# JURASSIC

Generated by Doxygen 1.8.11

ii CONTENTS

# Contents

1	Main	Page								2
2	Data	Structi	ıre Index							2
	2.1	Data S	tructures	 	 	 	 	 	 	2
3	File	Index								2
	3.1	File Lis	t	 	 	 	 	 	 	2
4	Data	Structi	ire Documentation							3
	4.1	atm_t s	Struct Reference	 	 	 	 	 	 	3
		4.1.1	Detailed Description	 	 	 	 	 	 	4
		4.1.2	Field Documentation	 	 	 	 	 	 	4
	4.2	ctl_t St	ruct Reference	 	 	 	 	 	 	5
		4.2.1	Detailed Description	 	 	 	 	 	 	6
		4.2.2	Field Documentation	 	 	 	 	 	 	6
	4.3	los_t S	truct Reference	 	 	 	 	 	 	9
		4.3.1	Detailed Description	 	 	 	 	 	 	10
		4.3.2	Field Documentation	 	 	 	 	 	 	10
	4.4	obs_t S	Struct Reference	 	 	 	 	 	 	12
		4.4.1	Detailed Description	 	 	 	 	 	 	12
		4.4.2	Field Documentation	 	 	 	 	 	 	12
	4.5	ret_t S	ruct Reference	 	 	 	 	 	 	14
		4.5.1	Detailed Description	 	 	 	 	 	 	15
		4.5.2	Field Documentation	 	 	 	 	 	 	15
	4.6	tbl_t St	ruct Reference	 	 	 	 	 	 	17
		4.6.1	Detailed Description	 	 	 	 	 	 	18
		4.6.2	Field Documentation	 	 	 	 	 	 	18

5	File	Documentation	19
	5.1	brightness.c File Reference	19
		5.1.1 Detailed Description	20
		5.1.2 Function Documentation	20
	5.2	brightness.c	20
	5.3	climatology.c File Reference	21
		5.3.1 Detailed Description	21
		5.3.2 Function Documentation	21
	5.4	climatology.c	22
	5.5	formod.c File Reference	23
		5.5.1 Detailed Description	23
		5.5.2 Function Documentation	23
	5.6	formod.c	27
	5.7	hydrostatic.c File Reference	30
		5.7.1 Detailed Description	30
		5.7.2 Function Documentation	30
	5.8	hydrostatic.c	31
	5.9	interpolate.c File Reference	32
		5.9.1 Detailed Description	32
		5.9.2 Function Documentation	32
	5.10	interpolate.c	33
	5.11	jsec2time.c File Reference	34
		5.11.1 Detailed Description	34
		5.11.2 Function Documentation	34
	5.12	? jsec2time.c	35
	5.13	g jurassic.c File Reference	35
		5.13.1 Detailed Description	37
		5.13.2 Function Documentation	37
	5.14	l jurassic.c	106
	5.15	jurassic.h File Reference	163

	5.15.1 Detailed Description	165
	5.15.2 Function Documentation	166
5.16	jurassic.h	234
5.17	kernel.c File Reference	241
	5.17.1 Detailed Description	241
	5.17.2 Function Documentation	241
5.18	kernel.c	243
5.19	limb.c File Reference	244
	5.19.1 Detailed Description	244
	5.19.2 Function Documentation	245
5.20	limb.c	245
5.21	nadir.c File Reference	246
	5.21.1 Detailed Description	246
	5.21.2 Function Documentation	247
5.22	nadir.c	247
5.23	planck.c File Reference	248
	5.23.1 Detailed Description	248
	5.23.2 Function Documentation	248
5.24	planck.c	249
5.25	raytrace.c File Reference	250
	5.25.1 Detailed Description	250
	5.25.2 Function Documentation	250
5.26	raytrace.c	251
5.27	retrieval.c File Reference	252
	5.27.1 Detailed Description	253
	5.27.2 Function Documentation	253
5.28	retrieval.c	265
5.29	time2jsec.c File Reference	276
	5.29.1 Detailed Description	276
	5.29.2 Function Documentation	277
5.30	time2jsec.c	277

Index 279

# 1 Main Page

The JUelich RApid Spectral SImulation Code (JURASSIC) is a fast radiative transfer model for the mid-infrared spectral region. This reference manual provides information on the algorithms and data structures used in the code. Further information can be found at: http://www.fz-juelich.de/ias/jsc/jurassic

# 2 Data Structure Index

## 2.1 Data Structures

Here are the data structures with brief descriptions:

Atmospheric data	3
ctl_t Forward model control parameters	Ę
los_t Line-of-sight data	ç
obs_t Observation geometry and radiance data	12
ret_t Retrieval control parameters	14
tbl_t Emissivity look-up tables	17

# 3 File Index

# 3.1 File List

Here is a list of all files with brief descriptions:

brightness.c Convert radiance to brightness temperature	19
climatology.c Prepare atmospheric data file from climatological data	21
formod.c  JURASSIC forward model	23
hydrostatic.c Recalculate pressure based on hydrostatic equilibrium	30

interpolate.c		
Interpolate atmospheric data to another	spatial grid	32
jsec2time.c		
Convert Julian seconds to date		34
jurassic.c		
JURASSIC library definitions		35
jurassic.h		163
JURASSIC library declarations		103
kernel.c Calculate kernel functions		241
		241
limb.c Create observation geometry for a limb s	sounder	244
nadir.c		
Create observation geometry for a nadir	sounder	246
planck.c		
Convert brightness temperature to radial	nce	248
raytrace.c		
Determine atmospheric ray paths		250
retrieval.c		
JURASSIC retrieval processor		252
time2jsec.c		
Convert date to Julian seconds		276
4 Data Structure Documentation		
4.1 atm_t Struct Reference		
Atmospheric data.		
<pre>#include <jurassic.h></jurassic.h></pre>		
Data Fields		
• int np		
Number of data points.		
double time [NP]		
Time (seconds since 2000-01-01T00:00Z).		
double z [NP]  Abitude [[mr.]]		
Altitude [km].  • double lon [NP]		
Longitude [deg].		
double lat [NP]		

Latitude [deg].

• double p [NP]

```
Pressure [hPa].
    • double t [NP]
           Temperature [K].

    double q [NG][NP]

          Volume mixing ratio.

    double k [NW][NP]

          Extinction [1/km].
4.1.1 Detailed Description
Atmospheric data.
Definition at line 206 of file jurassic.h.
4.1.2 Field Documentation
4.1.2.1 int atm_t::np
Number of data points.
Definition at line 209 of file jurassic.h.
4.1.2.2 double atm_t::time[NP]
Time (seconds since 2000-01-01T00:00Z).
Definition at line 212 of file jurassic.h.
4.1.2.3 double atm_t::z[NP]
Altitude [km].
Definition at line 215 of file jurassic.h.
4.1.2.4 double atm_t::lon[NP]
Longitude [deg].
Definition at line 218 of file jurassic.h.
4.1.2.5 double atm_t::lat[NP]
Latitude [deg].
Definition at line 221 of file jurassic.h.
4.1.2.6 double atm_t::p[NP]
Pressure [hPa].
Definition at line 224 of file jurassic.h.
```

```
4.1.2.7 double atm_t::t[NP]
Temperature [K].
Definition at line 227 of file jurassic.h.
4.1.2.8 double atm_t::q[NG][NP]
Volume mixing ratio.
Definition at line 230 of file jurassic.h.
4.1.2.9 double atm_t::k[NW][NP]
Extinction [1/km].
Definition at line 233 of file jurassic.h.
The documentation for this struct was generated from the following file:
    · jurassic.h
4.2 ctl_t Struct Reference
Forward model control parameters.
#include <jurassic.h>
Data Fields
    • int ng
          Number of emitters.
    • char emitter [NG][LEN]
          Name of each emitter.

 int nd

          Number of radiance channels.
    • int nw
          Number of spectral windows.
    • double nu [ND]
          Centroid wavenumber of each channel [cm^{\wedge}-1].
    • int window [ND]
          Window index of each channel.
    · char tblbase [LEN]
          Basename for table files and filter function files.
    · double hydz
          Reference height for hydrostatic pressure profile (-999 to skip) [km].
    • int ctm_co2
          Compute CO2 continuum (0=no, 1=yes).
    • int ctm h2o
          Compute H2O continuum (0=no, 1=yes).
```

· int ctm\_n2 Compute N2 continuum (0=no, 1=yes). • int ctm\_o2 Compute O2 continuum (0=no, 1=yes). · int refrac Take into account refractivity (0=no, 1=yes). · double rayds Maximum step length for raytracing [km]. · double raydz Vertical step length for raytracing [km]. char fov [LEN] Field-of-view data file. • double retp\_zmin Minimum altitude for pressure retrieval [km]. double retp zmax Maximum altitude for pressure retrieval [km]. double rett\_zmin Minimum altitude for temperature retrieval [km]. · double rett\_zmax Maximum altitude for temperature retrieval [km]. • double retq zmin [NG] Minimum altitude for volume mixing ratio retrieval [km]. double retq\_zmax [NG] Maximum altitude for volume mixing ratio retrieval [km]. • double retk\_zmin [NW] Minimum altitude for extinction retrieval [km]. double retk\_zmax [NW] Maximum altitude for extinction retrieval [km]. · int write bbt Use brightness temperature instead of radiance (0=no, 1=yes). · int write\_matrix Write matrix file (0=no, 1=yes). 4.2.1 Detailed Description Forward model control parameters. Definition at line 238 of file jurassic.h. 4.2.2 Field Documentation 4.2.2.1 int ctl\_t::ng Number of emitters.

Definition at line 241 of file jurassic.h.

4.2.2.2 char ctl\_t::emitter[NG][LEN] Name of each emitter. Definition at line 244 of file jurassic.h. 4.2.2.3 int ctl\_t::nd Number of radiance channels. Definition at line 247 of file jurassic.h. 4.2.2.4 int ctl\_t::nw Number of spectral windows. Definition at line 250 of file jurassic.h. 4.2.2.5 double ctl\_t::nu[ND] Centroid wavenumber of each channel [cm^-1]. Definition at line 253 of file jurassic.h. 4.2.2.6 int ctl\_t::window[ND] Window index of each channel. Definition at line 256 of file jurassic.h. 4.2.2.7 char ctl\_t::tblbase[LEN] Basename for table files and filter function files. Definition at line 259 of file jurassic.h. 4.2.2.8 double ctl\_t::hydz Reference height for hydrostatic pressure profile (-999 to skip) [km]. Definition at line 262 of file jurassic.h. 4.2.2.9 int ctl\_t::ctm\_co2 Compute CO2 continuum (0=no, 1=yes). Definition at line 265 of file jurassic.h. 4.2.2.10 int ctl\_t::ctm\_h2o Compute H2O continuum (0=no, 1=yes). Definition at line 268 of file jurassic.h.

```
4.2.2.11 int ctl_t::ctm_n2
Compute N2 continuum (0=no, 1=yes).
Definition at line 271 of file jurassic.h.
4.2.2.12 int ctl_t::ctm_o2
Compute O2 continuum (0=no, 1=yes).
Definition at line 274 of file jurassic.h.
4.2.2.13 int ctl_t::refrac
Take into account refractivity (0=no, 1=yes).
Definition at line 277 of file jurassic.h.
4.2.2.14 double ctl_t::rayds
Maximum step length for raytracing [km].
Definition at line 280 of file jurassic.h.
4.2.2.15 double ctl_t::raydz
Vertical step length for raytracing [km].
Definition at line 283 of file jurassic.h.
4.2.2.16 char ctl_t::fov[LEN]
Field-of-view data file.
Definition at line 286 of file jurassic.h.
4.2.2.17 double ctl_t::retp_zmin
Minimum altitude for pressure retrieval [km].
Definition at line 289 of file jurassic.h.
4.2.2.18 double ctl_t::retp_zmax
Maximum altitude for pressure retrieval [km].
Definition at line 292 of file jurassic.h.
4.2.2.19 double ctl_t::rett_zmin
Minimum altitude for temperature retrieval [km].
Definition at line 295 of file jurassic.h.
```

```
4.2.2.20 double ctl_t::rett_zmax
Maximum altitude for temperature retrieval [km].
Definition at line 298 of file jurassic.h.
4.2.2.21 double ctl_t::retq_zmin[NG]
Minimum altitude for volume mixing ratio retrieval [km].
Definition at line 301 of file jurassic.h.
4.2.2.22 double ctl_t::retq_zmax[NG]
Maximum altitude for volume mixing ratio retrieval [km].
Definition at line 304 of file jurassic.h.
4.2.2.23 double ctl_t::retk_zmin[NW]
Minimum altitude for extinction retrieval [km].
Definition at line 307 of file jurassic.h.
4.2.2.24 double ctl_t::retk_zmax[NW]
Maximum altitude for extinction retrieval [km].
Definition at line 310 of file jurassic.h.
4.2.2.25 int ctl_t::write_bbt
Use brightness temperature instead of radiance (0=no, 1=yes).
Definition at line 313 of file jurassic.h.
4.2.2.26 int ctl_t::write_matrix
Write matrix file (0=no, 1=yes).
Definition at line 316 of file jurassic.h.
The documentation for this struct was generated from the following file:
    • jurassic.h
4.3 los_t Struct Reference
Line-of-sight data.
#include <jurassic.h>
```

#### **Data Fields**

• int np

Number of LOS points.

double z [NLOS]

Altitude [km].

· double lon [NLOS]

Longitude [deg].

· double lat [NLOS]

Latitude [deg].

• double p [NLOS]

Pressure [hPa].

• double t [NLOS]

Temperature [K].

• double q [NG][NLOS]

Volume mixing ratio.

double k [NW][NLOS]

Extinction [1/km].

· double tsurf

Surface temperature [K].

· double ds [NLOS]

Segment length [km].

• double u [NG][NLOS]

Column density [molecules/cm<sup>2</sup>].

## 4.3.1 Detailed Description

Line-of-sight data.

Definition at line 321 of file jurassic.h.

4.3.2 Field Documentation

4.3.2.1 int los\_t::np

Number of LOS points.

Definition at line 324 of file jurassic.h.

4.3.2.2 double los\_t::z[NLOS]

Altitude [km].

Definition at line 327 of file jurassic.h.

4.3.2.3 double los\_t::lon[NLOS]

Longitude [deg].

Definition at line 330 of file jurassic.h.

```
4.3.2.4 double los_t::lat[NLOS]
Latitude [deg].
Definition at line 333 of file jurassic.h.
4.3.2.5 double los_t::p[NLOS]
Pressure [hPa].
Definition at line 336 of file jurassic.h.
4.3.2.6 double los_t::t[NLOS]
Temperature [K].
Definition at line 339 of file jurassic.h.
4.3.2.7 double los_t::q[NG][NLOS]
Volume mixing ratio.
Definition at line 342 of file jurassic.h.
4.3.2.8 double los_t::k[NW][NLOS]
Extinction [1/km].
Definition at line 345 of file jurassic.h.
4.3.2.9 double los_t::tsurf
Surface temperature [K].
Definition at line 348 of file jurassic.h.
4.3.2.10 double los_t::ds[NLOS]
Segment length [km].
Definition at line 351 of file jurassic.h.
4.3.2.11 double los_t::u[NG][NLOS]
Column density [molecules/cm<sup>2</sup>].
Definition at line 354 of file jurassic.h.
The documentation for this struct was generated from the following file:
    • jurassic.h
```

# 4.4 obs\_t Struct Reference

Observation geometry and radiance data.

```
#include <jurassic.h>
```

### **Data Fields**

• int nr

Number of ray paths.

• double time [NR]

Time (seconds since 2000-01-01T00:00Z).

• double obsz [NR]

Observer altitude [km].

• double obsion [NR]

Observer longitude [deg].

· double obslat [NR]

Observer latitude [deg].

double vpz [NR]

View point altitude [km].

• double vplon [NR]

View point longitude [deg].

double vplat [NR]

View point latitude [deg].

double tpz [NR]

Tangent point altitude [km].

• double tplon [NR]

Tangent point longitude [deg].

double tplat [NR]

Tangent point latitude [deg].

• double tau [ND][NR]

Transmittance of ray path.

· double rad [ND][NR]

Radiance [W/( $m^2$  sr cm $^-$ -1)].

#### 4.4.1 Detailed Description

Observation geometry and radiance data.

Definition at line 359 of file jurassic.h.

## 4.4.2 Field Documentation

## 4.4.2.1 int obs\_t::nr

Number of ray paths.

Definition at line 362 of file jurassic.h.

```
4.4.2.2 double obs_t::time[NR]
Time (seconds since 2000-01-01T00:00Z).
Definition at line 365 of file jurassic.h.
4.4.2.3 double obs_t::obsz[NR]
Observer altitude [km].
Definition at line 368 of file jurassic.h.
4.4.2.4 double obs_t::obslon[NR]
Observer longitude [deg].
Definition at line 371 of file jurassic.h.
4.4.2.5 double obs_t::obslat[NR]
Observer latitude [deg].
Definition at line 374 of file jurassic.h.
4.4.2.6 double obs_t::vpz[NR]
View point altitude [km].
Definition at line 377 of file jurassic.h.
4.4.2.7 double obs_t::vplon[NR]
View point longitude [deg].
Definition at line 380 of file jurassic.h.
4.4.2.8 double obs_t::vplat[NR]
View point latitude [deg].
Definition at line 383 of file jurassic.h.
4.4.2.9 double obs_t::tpz[NR]
Tangent point altitude [km].
Definition at line 386 of file jurassic.h.
4.4.2.10 double obs_t::tplon[NR]
Tangent point longitude [deg].
Definition at line 389 of file jurassic.h.
```

```
4.4.2.11 double obs_t::tplat[NR]
Tangent point latitude [deg].
Definition at line 392 of file jurassic.h.
4.4.2.12 double obs_t::tau[ND][NR]
Transmittance of ray path.
Definition at line 395 of file jurassic.h.
4.4.2.13 double obs_t::rad[ND][NR]
Radiance [W/(m^2 sr cm^--1)].
Definition at line 398 of file jurassic.h.
The documentation for this struct was generated from the following file:
    • jurassic.h
4.5
     ret_t Struct Reference
Retrieval control parameters.
Data Fields
    • char dir [LEN]
          Working directory.
    int kernel_recomp
          Recomputation of kernel matrix (number of iterations).
    · int conv itmax
          Maximum number of iterations.
    · double conv_dmin
          Minimum normalized step size in state space.
    · int err ana
          Carry out error analysis (0=no, 1=yes).

    double err_formod [ND]

          Forward model error [%].
    • double err_noise [ND]
          Noise error [W/(m^2 sr cm^--1)].

    double err_press

          Pressure error [%].
    • double err_press_cz
           Vertical correlation length for pressure error [km].
    double err_press_ch
          Horizontal correlation length for pressure error [km].

    double err_temp

           Temperature error [K].
```

```
    double err_temp_cz

           Vertical correlation length for temperature error [km].
     double err_temp_ch
           Horizontal correlation length for temperature error [km].

    double err_q [NG]

           Volume mixing ratio error [%].
     • double err_q_cz [NG]
           Vertical correlation length for volume mixing ratio error [km].
     double err_q_ch [NG]
           Horizontal correlation length for volume mixing ratio error [km].

    double err_k [NW]

           Extinction error [1/km].

    double err_k_cz [NW]

           Vertical correlation length for extinction error [km].

    double err_k_ch [NW]

           Horizontal correlation length for extinction error [km].
4.5.1 Detailed Description
Retrieval control parameters.
Definition at line 32 of file retrieval.c.
4.5.2 Field Documentation
4.5.2.1 char ret_t::dir[LEN]
Working directory.
Definition at line 35 of file retrieval.c.
4.5.2.2 int ret_t::kernel_recomp
Recomputation of kernel matrix (number of iterations).
Definition at line 38 of file retrieval.c.
4.5.2.3 int ret_t::conv_itmax
Maximum number of iterations.
Definition at line 41 of file retrieval.c.
4.5.2.4 double ret_t::conv_dmin
Minimum normalized step size in state space.
Definition at line 44 of file retrieval.c.
```

```
4.5.2.5 int ret_t::err_ana
Carry out error analysis (0=no, 1=yes).
Definition at line 47 of file retrieval.c.
4.5.2.6 double ret_t::err_formod[ND]
Forward model error [%].
Definition at line 50 of file retrieval.c.
4.5.2.7 double ret_t::err_noise[ND]
Noise error [W/(m^2 sr cm^--1)].
Definition at line 53 of file retrieval.c.
4.5.2.8 double ret_t::err_press
Pressure error [%].
Definition at line 56 of file retrieval.c.
4.5.2.9 double ret_t::err_press_cz
Vertical correlation length for pressure error [km].
Definition at line 59 of file retrieval.c.
4.5.2.10 double ret_t::err_press_ch
Horizontal correlation length for pressure error [km].
Definition at line 62 of file retrieval.c.
4.5.2.11 double ret_t::err_temp
Temperature error [K].
Definition at line 65 of file retrieval.c.
4.5.2.12 double ret_t::err_temp_cz
Vertical correlation length for temperature error [km].
Definition at line 68 of file retrieval.c.
4.5.2.13 double ret_t::err_temp_ch
Horizontal correlation length for temperature error [km].
Definition at line 71 of file retrieval.c.
```

```
4.5.2.14 double ret_t::err_q[NG]
Volume mixing ratio error [%].
Definition at line 74 of file retrieval.c.
4.5.2.15 double ret_t::err_q_cz[NG]
Vertical correlation length for volume mixing ratio error [km].
Definition at line 77 of file retrieval.c.
4.5.2.16 double ret_t::err_q_ch[NG]
Horizontal correlation length for volume mixing ratio error [km].
Definition at line 80 of file retrieval.c.
4.5.2.17 double ret_t::err_k[NW]
Extinction error [1/km].
Definition at line 83 of file retrieval.c.
4.5.2.18 double ret_t::err_k_cz[NW]
Vertical correlation length for extinction error [km].
Definition at line 86 of file retrieval.c.
4.5.2.19 double ret_t::err_k_ch[NW]
Horizontal correlation length for extinction error [km].
Definition at line 89 of file retrieval.c.
The documentation for this struct was generated from the following file:
    · retrieval.c
4.6 tbl_t Struct Reference
Emissivity look-up tables.
```

#include <jurassic.h>

#### **Data Fields**

• int np [NG][ND]

Number of pressure levels.

• int nt [NG][ND][TBLNP]

Number of temperatures.

• int nu [NG][ND][TBLNP][TBLNT]

Number of column densities.

• double p [NG][ND][TBLNP]

Pressure [hPa].

• double t [NG][ND][TBLNP][TBLNT]

Temperature [K].

• float u [NG][ND][TBLNP][TBLNT][TBLNU]

Column density [molecules/cm<sup>2</sup>].

float eps [NG][ND][TBLNP][TBLNT][TBLNU]

Emissivity.

• double st [TBLNS]

Source function temperature [K].

• double sr [ND][TBLNS]

Source function radiance [W/( $m^2$  sr cm $^-$ -1)].

#### 4.6.1 Detailed Description

Emissivity look-up tables.

Definition at line 403 of file jurassic.h.

4.6.2 Field Documentation

4.6.2.1 int tbl\_t::np[NG][ND]

Number of pressure levels.

Definition at line 406 of file jurassic.h.

4.6.2.2 int tbl\_t::nt[NG][ND][TBLNP]

Number of temperatures.

Definition at line 409 of file jurassic.h.

4.6.2.3 int tbl\_t::nu[NG][ND][TBLNP][TBLNT]

Number of column densities.

Definition at line 412 of file jurassic.h.

5 File Documentation 19

```
4.6.2.4 double tbl_t::p[NG][ND][TBLNP]
Pressure [hPa].
Definition at line 415 of file jurassic.h.
4.6.2.5 double tbl_t::t[NG][ND][TBLNP][TBLNT]
Temperature [K].
Definition at line 418 of file jurassic.h.
4.6.2.6 float tbl_t::u[NG][ND][TBLNP][TBLNT][TBLNU]
Column density [molecules/cm<sup>2</sup>].
Definition at line 421 of file jurassic.h.
4.6.2.7 float tbl_t::eps[NG][ND][TBLNP][TBLNT][TBLNU]
Emissivity.
Definition at line 424 of file jurassic.h.
4.6.2.8 double tbl_t::st[TBLNS]
Source function temperature [K].
Definition at line 427 of file jurassic.h.
4.6.2.9 double tbl_t::sr[ND][TBLNS]
Source function radiance [W/(m^2 sr cm^--1)].
Definition at line 430 of file jurassic.h.
The documentation for this struct was generated from the following file:
    • jurassic.h
    File Documentation
5.1 brightness.c File Reference
Convert radiance to brightness temperature.
Functions
    • int main (int argc, char *argv[])
```

#### 5.1.1 Detailed Description

Convert radiance to brightness temperature.

Definition in file brightness.c.

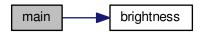
#### 5.1.2 Function Documentation

#### 5.1.2.1 int main (int argc, char \* argv[])

Definition at line 27 of file brightness.c.

```
00029
00030
00031
        double nu, rad;
00032
00033
        /* Check arguments... */
00034
        if (argc < 3)
00035
          ERRMSG("Give parameters: <rad> <nu>");
00036
00037
        /* Read arguments... */
00038
       rad = atof(argv[1]);
00039
       nu = atof(argv[2]);
00040
00041
        /* Compute brightness temperature... */
00042
        printf("%.10g\n", brightness(rad, nu));
00043
00044
        return EXIT_SUCCESS;
00045 }
```

Here is the call graph for this function:



# 5.2 brightness.c

```
00001 /*
00002
         This file is part of JURASSIC.
00003
00004
         {\tt JURASSIC} is free software: you can redistribute it and/or modify
         it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
00005
00006
00007
         (at your option) any later version.
80000
00009
          JURASSIC is distributed in the hope that it will be useful,
         but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00010
00011
00012
         GNU General Public License for more details.
00013
00014
         You should have received a copy of the GNU General Public License
00015
         along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
         Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
```

```
00027 int main(
00028
       int argc,
00029
       char *argv[]) {
00030
00031
       double nu, rad;
00032
00033
       /* Check arguments... */
00034
        if (argc < 3)
00035
        ERRMSG("Give parameters: <rad> <nu>");
00036
00037
       /* Read arguments... */
00038
       rad = atof(argv[1]);
00039
       nu = atof(argv[2]);
00040
00041
       /\star Compute brightness temperature... \star/
00042
       printf("%.10g\n", brightness(rad, nu));
00043
00044
       return EXIT_SUCCESS;
00045 }
```

## 5.3 climatology.c File Reference

Prepare atmospheric data file from climatological data.

#### **Functions**

• int main (int argc, char \*argv[])

#### 5.3.1 Detailed Description

Prepare atmospheric data file from climatological data.

Definition in file climatology.c.

#### 5.3.2 Function Documentation

## 5.3.2.1 int main ( int argc, char \* argv[])

Definition at line 27 of file climatology.c.

```
00029
00030
00031
          static atm_t atm;
00032
          static ctl_t ctl;
00033
00034
          double dz, t0, z, z0, z1;
00035
00036
          /* Check arguments... ∗/
00037
          if (argc < 3)</pre>
00038
            ERRMSG("Give parameters: <ctl> <atm>");
00039
00040
          /* Read control parameters... */
00041
          read_ctl(argc, argv, &ctl);
         t0 = scan_ctl(argc, argv, "T0", -1, "0", NULL);

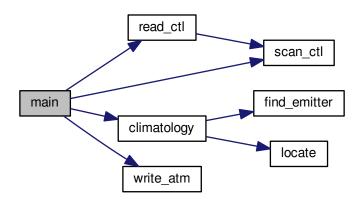
z0 = scan_ctl(argc, argv, "Z0", -1, "0", NULL);

z1 = scan_ctl(argc, argv, "Z1", -1, "90", NULL);

dz = scan_ctl(argc, argv, "DZ", -1, "1", NULL);
00042
00043
00044
00045
00046
00047
          /* Set atmospheric grid... */
00048
          for (z = z0; z \le z1; z += dz) {
          atm.time[atm.np] = t0;
00049
00050
            atm.z[atm.np] = z;
if ((++atm.np) >= NP)
00051
00052
               ERRMSG("Too many atmospheric grid points!");
00053
```

```
00054
00055    /* Interpolate climatological data... */
00056    climatology(&ctl, &atm);
00057
00058    /* Write data to disk... */
00059    write_atm(NULL, argv[2], &ctl, &atm);
00060
00061    return EXIT_SUCCESS;
00062 }
```

Here is the call graph for this function:



# 5.4 climatology.c

```
00001 /*
00002
         This file is part of JURASSIC.
00003
00004
         JURASSIC is free software: you can redistribute it and/or modify
         it under the terms of the GNU General Public License as published by
00005
00006
         the Free Software Foundation, either version 3 of the License, or
00007
         (at your option) any later version.
00008
         JURASSIC is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00009
00010
00011
00012
         GNU General Public License for more details.
00013
00014
         You should have received a copy of the GNU General Public License
00015
         along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
         Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028
         int argc,
00029
         char *argv[]) {
00030
         static atm_t atm;
static ctl_t ctl;
00031
00032
00033
00034
         double dz, t0, z, z0, z1;
00035
00036
         /* Check arguments... */
00037
         if (argc < 3)
           ERRMSG("Give parameters: <ctl> <atm>");
00038
00039
00040
         /* Read control parameters... */
         read_ctl(argc, argv, &ctl);
t0 = scan_ctl(argc, argv, "T0", -1, "0", NULL);
00041
```

```
z0 = scan_ctl(argc, argv, "Z0", -1, "0", NULL);
z1 = scan_ctl(argc, argv, "Z1", -1, "90", NULL);
dz = scan_ctl(argc, argv, "DZ", -1, "1", NULL);
00045
00046
00047
         /* Set atmospheric grid... */
00048
         for (z = z0; z \le z1; z += dz) {
          atm.time[atm.np] = t0;
00050
           atm.z[atm.np] = z;
00051
           if ((++atm.np) >= NP)
00052
              ERRMSG("Too many atmospheric grid points!");
00053
00054
00055
         /* Interpolate climatological data... */
00056
        climatology(&ctl, &atm);
00057
         /* Write data to disk... */
write_atm(NULL, argv[2], &ctl, &atm);
00058
00059
00060
00061
         return EXIT_SUCCESS;
00062 }
```

#### 5.5 formod.c File Reference

JURASSIC forward model.

#### **Functions**

void call\_formod (ctl\_t \*ctl, const char \*wrkdir, const char \*obsfile, const char \*atmfile, const char \*radfile, const char \*task)

Perform forward model calculations in a single directory.

• int main (int argc, char \*argv[])

#### 5.5.1 Detailed Description

JURASSIC forward model.

Definition in file formod.c.

#### 5.5.2 Function Documentation

5.5.2.1 void call\_formod ( ctl\_t \* ctl, const char \* wrkdir, const char \* obsfile, const char \* atmfile, const char \* radfile, const char \* task )

Perform forward model calculations in a single directory.

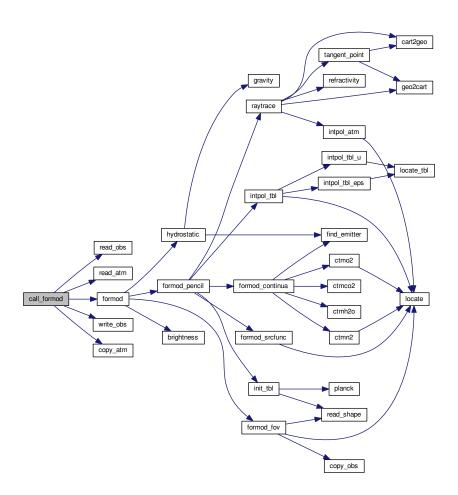
Definition at line 97 of file formod.c.

```
00103
                          {
00104
00105
        static atm_t atm, atm2;
00106
       static obs_t obs, obs2;
00107
00108
       char filename[LEN];
00109
00110
       int id, ig, ig2, ip, ir, iw;
00111
00112
        /* Read observation geometry... */
00113
       read_obs(wrkdir, obsfile, ctl, &obs);
00114
00115
       /* Read atmospheric data... */
00116
       read_atm(wrkdir, atmfile, ctl, &atm);
00117
```

```
/* Compute multiple profiles... */
if (task[0] == 'p' || task[0] == 'P') {
00118
00119
00120
            /* Loop over ray paths... */
for (ir = 0; ir < obs.nr; ir++) {</pre>
00121
00122
00123
00124
               /* Get atmospheric data... */
00125
               atm2.np = 0;
               for (ip = 0; ip < atm.np; ip++)</pre>
00126
                 if (atm.time[ip] == obs.time[ir]) {
  atm2.time[atm2.np] = atm.time[ip];
00127
00128
                   atm2.z[atm2.np] = atm.z[ip];
atm2.lon[atm2.np] = atm.lon[ip];
atm2.lat[atm2.np] = atm.lat[ip];
00129
00130
00131
                   atm2.p[atm2.np] = atm.p[ip];
atm2.t[atm2.np] = atm.t[ip];
for (ig = 0; ig < ctl->ng; ig++)
00132
00133
00134
                    atm2.q[ig][atm2.np] = atm.q[ig][ip];
for (iw = 0; iw < ctl->nw; iw++)
00135
00136
                      atm2.k[iw][atm2.np] = atm.k[iw][ip];
00137
00138
                    atm2.np++;
00139
00140
00141
               /* Get observation data... */
00142
               obs2.nr = 1;
               obs2.time[0] = obs.time[ir];
00144
               obs2.vpz[0] = obs.vpz[ir];
              obs2.vplon[0] = obs.vplon[ir];
obs2.vplat[0] = obs.vplat[ir];
00145
00146
00147
               obs2.obsz[0] = obs.obsz[ir];
              obs2.obslon[0] = obs.obslon[ir];
00148
00149
              obs2.obslat[0] = obs.obslat[ir];
00150
00151
               /\star Check number of data points... \star/
00152
              if (atm2.np > 0) {
00153
                 /* Call forward model... */
00154
00155
                 formod(ctl, &atm2, &obs2);
00156
00157
                  /* Save radiance data... */
                 for (id = 0; id < ctl->nd; id++) {
  obs.rad[id][ir] = obs2.rad[id][0];
  obs.tau[id][ir] = obs2.tau[id][0];
00158
00159
00160
00161
                 }
00162
00163
00164
00165
             /* Write radiance data... */
            write_obs(wrkdir, radfile, ctl, &obs);
00166
00167
00168
00169
          /* Compute single profile... */
00170
          else {
00171
00172
            /* Call forward model... */
00173
            formod(ctl, &atm, &obs);
00175
            /* Save radiance data... */
00176
            write_obs(wrkdir, radfile, ctl, &obs);
00177
00178
            /* Compute contributions... */
            if (task[0] == 'c' || task[0] == 'C') {
00179
00180
00181
               /* Switch off continua... */
00182
              ct1->ctm\_co2 = 0;
00183
               ct1->ctm_h2o = 0;
              ctl->ctm_n2 = 0:
00184
00185
              ct1->ctm o2 = 0;
00186
               /* Loop over emitters... */
00188
               for (ig = 0; ig < ctl->ng; ig++) {
00189
                 /* Copy atmospheric data... */
copy_atm(ctl, &atm2, &atm, 0);
00190
00191
00192
00193
                  /* Set extinction to zero... */
00194
                 for (iw = 0; iw < ctl->nw; iw++)
00195
                   for (ip = 0; ip < atm2.np; ip++)</pre>
00196
                      atm2.k[iw][ip] = 0;
00197
00198
                  /\star Set volume mixing ratios to zero... \star/
                 for (ig2 = 0; ig2 < ct1->ng; ig2++)
00199
00200
                    if (ig2 != ig)
00201
                      for (ip = 0; ip < atm2.np; ip++)</pre>
00202
                        atm2.q[ig2][ip] = 0;
00203
00204
                 /* Call forward model... */
```

```
formod(ctl, &atm2, &obs);
00206
                    /* Save radiance data... */
sprintf(filename, "%s.%s", radfile, ctl->emitter[ig]);
write_obs(wrkdir, filename, ctl, &obs);
00207
00208
00209
00210
00211
00212
                  /\star Copy atmospheric data... \star/
00213
                 copy_atm(ctl, &atm2, &atm, 0);
00214
                 /* Set volume mixing ratios to zero... */
for (ig = 0; ig < ctl->ng; ig++)
    for (ip = 0; ip < atm2.np; ip++)
        atm2.q[ig][ip] = 0;</pre>
00215
00216
00217
00218
00219
00220
                 /* Call forward model... */
00221
                 formod(ctl, &atm2, &obs);
00222
00223
                 /* Save radiance data... */
                 sprintf(filename, "%s.EXTINCT", radfile);
write_obs(wrkdir, filename, ctl, &obs);
00224
00225
00226
00227
              /* Measure CPU-time... */
if (task[0] == 't' || task[0] == 'T') {
    TIMER("formod", 1);
00228
00229
00230
00231
                  formod(ctl, &atm, &obs);
00232
                TIMER("formod", 3);
00233
00234 }
00235 }
```

Here is the call graph for this function:



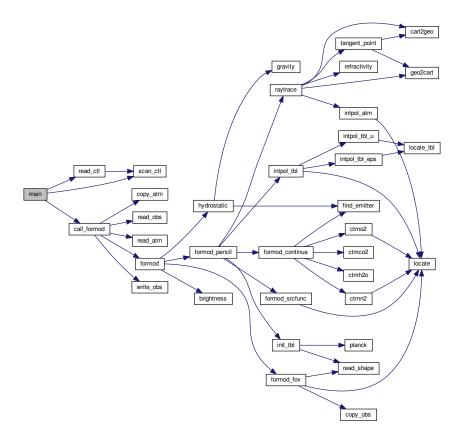
5.5.2.2 int main (int argc, char \* argv[])

Definition at line 44 of file formod.c.

```
00046
                        {
00047
00048
        static ctl_t ctl;
00049
00050
00051
00052
        char dirlist[LEN], task[LEN], wrkdir[LEN];
00053
00054
        /* Check arguments... */
00055
         if (argc < 5)
00056
           ERRMSG("Give parameters: <ctl> <obs> <atm> <rad>");
00057
00058
        /* Read control parameters... */
00059
        read_ctl(argc, argv, &ctl);
00060
00061
        /* Get task... */
00062
        scan_ctl(argc, argv, "TASK", -1, "-", task);
00063
00064
        /* Get dirlist... */
        scan_ctl(argc, argv, "DIRLIST", -1, "-", dirlist);
00065
00066
00067
         /* Single forward calculation... */
00068
         if (dirlist[0] == '-')
00069
           call_formod(&ctl, NULL, argv[2], argv[3], argv[4], task);
00070
00071
        /* Work on directory list... */
00072
        else {
00073
          /* Open directory list... */
if (!(in = fopen(dirlist, "r")))
    ERRMSG("Cannot open directory list!");
00074
00075
00076
00077
00078
           /* Loop over directories... */
00079
           while (fscanf(in, "%s", wrkdir) != EOF) {
08000
             /* Write info... */ printf("\nWorking directory: s\n", wrkdir);
00081
00082
00083
             /* Call forward model... */
call_formod(&ctl, wrkdir, argv[2], argv[3], argv[4], task);
00084
00085
00086
00087
00088
           /* Close dirlist... */
00089
          fclose(in);
00090
00091
00092
        return EXIT_SUCCESS;
00093 }
```

5.6 formod.c 27

Here is the call graph for this function:



#### 5.6 formod.c

```
00001 /*
00002
        This file is part of JURASSIC.
00003
00004
         {\tt JURASSIC} is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
00005
        the Free Software Foundation, either version 3 of the License, or
00006
00007
        (at your option) any later version.
80000
00009
        JURASSIC is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00010
00011
00012
        GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 /*
00028
         Functions...
00029
00030
00032 void call_formod(
00033
        ctl_t * ctl,
00034
        const char *wrkdir,
00035
         const char *obsfile,
00036
        const char *atmfile,
00037
        const char *radfile,
00038
        const char *task);
00039
00040 /*
```

```
00041
         Main...
00042
00043
00044 int main(
00045
        int argc,
00046
        char *argv[]) {
00047
00048
        static ctl_t ctl;
00049
00050
       FILE *in;
00051
00052
        char dirlist[LEN], task[LEN], wrkdir[LEN];
00053
00054
        /* Check arguments... */
00055
        if (argc < 5)
00056
          ERRMSG("Give parameters: <ctl> <obs> <atm> <rad>");
00057
00058
        /* Read control parameters... */
00059
        read_ctl(argc, argv, &ctl);
00060
        /* Get task... */
00061
        scan_ctl(argc, argv, "TASK", -1, "-", task);
00062
00063
00064
        /* Get dirlist... */
00065
        scan_ctl(argc, argv, "DIRLIST", -1, "-", dirlist);
00066
00067
        /\star Single forward calculation... \star/
00068
        if (dirlist[0] == '-')
          call_formod(&ctl, NULL, argv[2], argv[3], argv[4], task);
00069
00070
00071
        /* Work on directory list... */
00072
        else {
00073
00074
           /\star Open directory list... \star/
          if (!(in = fopen(dirlist, "r")))
    ERRMSG("Cannot open directory list!");
00075
00076
00077
00078
          /* Loop over directories... */
00079
          while (fscanf(in, "%s", wrkdir) != EOF) {
00080
            /* Write info... */
printf("\nWorking directory: %s\n", wrkdir);
00081
00082
00083
00084
             /* Call forward model... */
00085
            call_formod(&ctl, wrkdir, argv[2], argv[3], argv[4], task);
00086
00087
00088
           /* Close dirlist... */
00089
          fclose(in);
00090
00091
00092
        return EXIT_SUCCESS;
00093 }
00094
00096
00097 void call_formod(
00098
        ctl_t * ctl,
00099
        const char *wrkdir,
00100
        const char *obsfile,
00101
        const char *atmfile,
00102
        const char *radfile,
00103
        const char *task) {
00104
00105
        static atm_t atm, atm2;
00106
       static obs_t obs, obs2;
00107
00108
        char filename[LEN];
00109
00110
        int id, ig, ig2, ip, ir, iw;
00111
00112
        /* Read observation geometry... */
00113
        read_obs(wrkdir, obsfile, ctl, &obs);
00114
        /* Read atmospheric data... */
00115
00116
        read_atm(wrkdir, atmfile, ctl, &atm);
00117
        /* Compute multiple profiles... */
if (task[0] == 'p' || task[0] == 'P') {
00118
00119
00120
          /* Loop over ray paths... */
for (ir = 0; ir < obs.nr; ir++) {</pre>
00121
00122
00123
00124
             /* Get atmospheric data... */
            atm2.np = 0;
for (ip = 0; ip < atm.np; ip++)
  if (atm.time[ip] == obs.time[ir]) {</pre>
00125
00126
00127
```

5.6 formod.c 29

```
atm2.time[atm2.np] = atm.time[ip];
                   atm2.z[atm2.np] = atm.z[ip];
atm2.lon[atm2.np] = atm.lon[ip];
atm2.lat[atm2.np] = atm.lat[ip];
00129
00130
00131
00132
                   atm2.p[atm2.np] = atm.p[ip];
atm2.t[atm2.np] = atm.t[ip];
00133
                   for (ig = 0; ig < ctl->ng; ig++)
00134
00135
                      atm2.q[ig][atm2.np] = atm.q[ig][ip];
00136
                   for (iw = 0; iw < ctl->nw; iw++)
00137
                     atm2.k[iw][atm2.np] = atm.k[iw][ip];
00138
                   atm2.np++;
00139
00140
00141
               /* Get observation data... */
00142
               obs2.nr = 1;
              obs2.time[0] = obs.time[ir];
obs2.vpz[0] = obs.vpz[ir];
obs2.vplon[0] = obs.vplon[ir];
obs2.vplat[0] = obs.vplat[ir];
00143
00144
00145
              obs2.obsz[0] = obs.obsz[ir];
00147
              obs2.obslon[0] = obs.obslon[ir];
obs2.obslat[0] = obs.obslat[ir];
00148
00149
00150
              /\star Check number of data points... \star/
00151
              if (atm2.np > 0) {
00152
00153
00154
                 /\star Call forward model... \star/
00155
                formod(ctl, &atm2, &obs2);
00156
00157
                 /* Save radiance data... */
                 for (id = 0; id < ctl->nd; id++) {
  obs.rad[id][ir] = obs2.rad[id][0];
00158
00159
00160
                   obs.tau[id][ir] = obs2.tau[id][0];
00161
00162
              }
00163
00164
00165
            /* Write radiance data... */
00166
            write_obs(wrkdir, radfile, ctl, &obs);
00167
00168
         /* Compute single profile... */
00169
00170
         else (
00171
00172
            /* Call forward model... */
00173
            formod(ctl, &atm, &obs);
00174
00175
            /* Save radiance data... */
            write_obs(wrkdir, radfile, ctl, &obs);
00176
00177
00178
            /* Compute contributions... */
00179
            if (task[0] == 'c' || task[0] == 'C') {
00180
00181
               /* Switch off continua... */
00182
              ct1->ctm_co2 = 0;
              ctl \rightarrow ctm_h2o = 0;
00183
              ct1->ctm_n^2 = 0;
00185
              ct1->ctm_o2 = 0;
00186
00187
               /* Loop over emitters... */
00188
              for (ig = 0; ig < ctl->ng; ig++) {
00189
00190
                 /* Copy atmospheric data...
00191
                 copy_atm(ctl, &atm2, &atm, 0);
00192
00193
                 /* Set extinction to zero... */
00194
                 for (iw = 0; iw < ctl->nw; iw++)
  for (ip = 0; ip < atm2.np; ip++)</pre>
00195
00196
                     atm2.k[iw][ip] = 0;
00197
00198
                 /* Set volume mixing ratios to zero... */
00199
                 for (ig2 = 0; ig2 < ct1->ng; ig2++)
                   if (ig2 != ig)
   for (ip = 0; ip < atm2.np; ip++)</pre>
00200
00201
                        atm2.q[ig2][ip] = 0;
00202
00203
00204
                 /* Call forward model... */
00205
                 formod(ct1, &atm2, &obs);
00206
                /* Save radiance data... */
sprintf(filename, "%s.%s", radfile, ctl->emitter[ig]);
write_obs(wrkdir, filename, ctl, &obs);
00207
00208
00209
00210
00211
00212
              /* Copy atmospheric data... */
00213
              copy_atm(ctl, &atm2, &atm, 0);
00214
```

```
/* Set volume mixing ratios to zero... */
              for (ig = 0; ig < ctl->ng; ig++)
for (ip = 0; ip < atm2.np; ip++)
00216
00217
                   atm2.q[ig][ip] = 0;
00218
00219
00220
              /* Call forward model... */
00221
              formod(ct1, &atm2, &obs);
00222
00223
               /* Save radiance data... */
              sprintf(filename, "%s.EXTINCT", radfile);
write_obs(wrkdir, filename, ctl, &obs);
00224
00225
00226
00227
00228
            /* Measure CPU-time... */
           if (task[0] == 't' || task[0] == 'T') {
    TIMER("formod", 1);
00229
00230
              formod(ctl, &atm, &obs);
TIMER("formod", 3);
00231
00232
00233
00234
         }
00235 }
```

## 5.7 hydrostatic.c File Reference

Recalculate pressure based on hydrostatic equilibrium.

#### **Functions**

• int main (int argc, char \*argv[])

#### 5.7.1 Detailed Description

Recalculate pressure based on hydrostatic equilibrium.

Definition in file hydrostatic.c.

### 5.7.2 Function Documentation

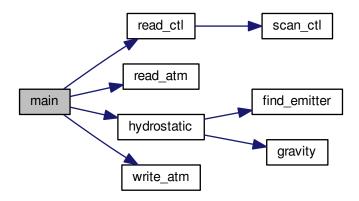
#### 5.7.2.1 int main (int argc, char \* argv[])

Definition at line 27 of file hydrostatic.c.

```
00030
00031
        static atm_t atm;
00032
       static ctl_t ctl;
00033
00034
        /* Check arguments... */
00035
       if (argc < 4)
          ERRMSG("Give parameters: <ctl> <atm_in> <atm_hyd>");
00037
00038
       /* Read control parameters... ∗/
00039
       read_ctl(argc, argv, &ctl);
00040
00041
        /* Check reference height... */
       if (ctl.hydz < 0)
00042
00043
          ERRMSG("Set HYDZ>=0!");
0\,0\,0\,4\,4
00045
       /* Read atmospheric data... */
00046
       read_atm(NULL, argv[2], &ctl, &atm);
00047
00048
        /\star Build atmosphere based on hydrostatic equilibrium... \star/
00049
       hydrostatic(&ctl, &atm);
00050
00051
        /* Write atmospheric data... */
00052
       write_atm(NULL, argv[3], &ctl, &atm);
00053
00054
        return EXIT_SUCCESS;
00055 }
```

5.8 hydrostatic.c 31

Here is the call graph for this function:



## 5.8 hydrostatic.c

```
00001 /*
00002
        This file is part of JURASSIC.
00003
        JURASSIC is free software: you can redistribute it and/or modify
00004
         it under the terms of the GNU General Public License as published by
00005
00006
        the Free Software Foundation, either version 3 of the License, or
00007
         (at your option) any later version.
80000
        JURASSIC is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00009
00010
00011
00012
        GNU General Public License for more details.
00013
00014
         You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028
        int argc,
00029
        char *argv[]) {
00030
00031
        static atm_t atm;
00032
        static ctl_t ctl;
00033
00034
        /* Check arguments... */
00035
        if (argc < 4)
00036
          ERRMSG("Give parameters: <ctl> <atm_in> <atm_hyd>");
00037
00038
        /* Read control parameters... */
00039
        read_ctl(argc, argv, &ctl);
00040
00041
         /* Check reference height... */
00042
        if (ctl.hydz < 0)</pre>
00043
          ERRMSG("Set HYDZ>=0!");
0\,0\,0\,4\,4
00045
        /* Read atmospheric data... */
00046
        read_atm(NULL, argv[2], &ctl, &atm);
00047
00048
         /* Build atmosphere based on hydrostatic equilibrium... */
00049
        hydrostatic(&ctl, &atm);
00050
00051
         /* Write atmospheric data... */
00052
        write_atm(NULL, argv[3], &ctl, &atm);
00053
00054
        return EXIT_SUCCESS;
00055 }
```

# 5.9 interpolate.c File Reference

Interpolate atmospheric data to another spatial grid.

#### **Functions**

• int main (int argc, char \*argv[])

#### 5.9.1 Detailed Description

Interpolate atmospheric data to another spatial grid.

Definition in file interpolate.c.

#### 5.9.2 Function Documentation

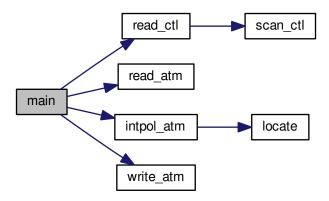
```
5.9.2.1 int main (int argc, char * argv[])
```

Definition at line 27 of file interpolate.c.

```
00029
00030
        static atm_t atm_in, atm_pts;
static ctl_t ctl;
00031
00032
00033
00034
        double k[NW], q[NG];
00035
00036
        int ig, ip, iw;
00037
00038
        /* Interpolate atmospheric data... */
00039
00040
        /* Check arguments... */
00041
00042
         ERRMSG("Give parameters: <ctl> <atm_in> <atm_pts> <atm_out>");
00043
00044
        /* Read control parameters... */
00045
        read_ctl(argc, argv, &ctl);
00046
00047
        /* Read atmospheric data... */
00048
        read_atm(NULL, argv[2], &ctl, &atm_in);
00049
        read_atm(NULL, argv[3], &ctl, &atm_pts);
00050
00051
        /* Interpolate atmospheric data... */
        for (ip = 0; ip < atm_pts.np; ip++) {</pre>
00052
00053
         intpol_atm(&ctl, &atm_in, atm_pts.z[ip],
00054
                      &atm_pts.p[ip], &atm_pts.t[ip], q, k);
         for (ig = 0; ig < ctl.ng; ig++)
00055
         atm_pts.q[ig][ip] = q[ig];
for (iw = 0; iw < ctl.nw; iw++)
00056
00057
            atm_pts.k[iw][ip] = k[iw];
00058
00059
00060
00061
00062
        /* Save interpolated data... */
        write_atm(NULL, argv[4], &ctl, &atm_pts);
00063
00064
        return EXIT_SUCCESS;
00065 }
```

5.10 interpolate.c 33

Here is the call graph for this function:



# 5.10 interpolate.c

```
00001 /*
00002
        This file is part of {\tt JURASSIC}.
00003
00004
        JURASSIC is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
00006
        the Free Software Foundation, either version 3 of the License, or
00007
        (at your option) any later version.
00008
00009
        JURASSIC is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
00010
00011
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00012
        GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028
        int argc,
00029
        char *argv[]) {
00030
00031
        static atm_t atm_in, atm_pts;
00032
        static ctl_t ctl;
00033
00034
       double k[NW], q[NG];
00035
00036
        int ig, ip, iw;
00037
00038
        /* Interpolate atmospheric data... */
00039
00040
        /* Check arguments... */
00041
        if (argc < 5)
00042
          ERRMSG("Give parameters: <ctl> <atm_in> <atm_pts> <atm_out>");
00043
00044
        /* Read control parameters... */
00045
        read_ctl(argc, argv, &ctl);
00046
00047
        /* Read atmospheric data... */
00048
        read_atm(NULL, argv[2], &ctl, &atm_in);
00049
        read_atm(NULL, argv[3], &ctl, &atm_pts);
00050
        /* Interpolate atmospheric data... */
for (ip = 0; ip < atm_pts.np; ip++) {</pre>
00051
00052
         00053
00054
```

```
for (ig = 0; ig < ctl.ng; ig++)</pre>
          atm_pts.q[ig][ip] = q[ig];
for (iw = 0; iw < ctl.nw; iw++)
00056
00057
00058
           atm_pts.k[iw][ip] = k[iw];
00059
00060
        /* Save interpolated data... */
00061
00062
        write_atm(NULL, argv[4], &ctl, &atm_pts);
00063
00064
        return EXIT_SUCCESS;
00065 }
```

# 5.11 jsec2time.c File Reference

Convert Julian seconds to date.

## **Functions**

• int main (int argc, char \*argv[])

# 5.11.1 Detailed Description

Convert Julian seconds to date.

Definition in file jsec2time.c.

## 5.11.2 Function Documentation

# 5.11.2.1 int main ( int argc, char \* argv[] )

Definition at line 27 of file jsec2time.c.

```
00029
00030
00031
        double jsec, remain;
00032
00033
         int day, hour, min, mon, sec, year;
00035
         /* Check arguments... */
00036
           ERRMSG("Give parameters: <jsec>");
00037
00038
00039
        /* Read arguments... */
00040
        jsec = atof(argv[1]);
00041
00042
         /* Convert time... */
         jsec2time(jsec, &year, &mon, &day, &hour, &min, &sec, &remain);
printf("%d %d %d %d %d %d %g\n", year, mon, day, hour, min, sec, remain);
00043
00044
00045
00046
         return EXIT_SUCCESS;
00047 }
```

Here is the call graph for this function:



5.12 jsec2time.c 35

## 5.12 jsec2time.c

```
00001 /*
00002
        This file is part of JURASSIC.
00003
00004
        JURASSIC is free software: you can redistribute it and/or modify
00005
        it under the terms of the GNU General Public License as published by
        the Free Software Foundation, either version 3 of the License, or
00006
00007
        (at your option) any later version.
80000
00009
        JURASSIC is distributed in the hope that it will be useful,
00010
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00011
00012
        GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00017
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028
        int argc,
00029
        char *argv[]) {
00030
        double jsec, remain;
00032
00033
        int day, hour, min, mon, sec, year;
00034
00035
        /* Check arguments... */
00036
        if (argc < 2)
00037
          ERRMSG("Give parameters: <jsec>");
00038
00039
        /* Read arguments.
00040
        jsec = atof(argv[1]);
00041
00042
        /* Convert time... */
00043
        jsec2time(jsec, &year, &mon, &day, &hour, &min, &sec, &remain);
00044
        printf("%d %d %d %d %d %d %g\n", year, mon, day, hour, min, sec, remain);
00045
00046
        return EXIT_SUCCESS;
00047 }
```

## 5.13 jurassic.c File Reference

JURASSIC library definitions.

#### **Functions**

size\_t atm2x (ctl\_t \*ctl, atm\_t \*atm, gsl\_vector \*x, int \*iqa, int \*ipa)

Compose state vector or parameter vector.

void atm2x\_help (atm\_t \*atm, double zmin, double zmax, double \*value, int val\_iqa, gsl\_vector \*x, int \*iqa, int \*ipa, size t \*n)

Add elements to state vector.

• double brightness (double rad, double nu)

Compute brightness temperature.

void cart2geo (double \*x, double \*z, double \*lon, double \*lat)

Convert Cartesian coordinates to geolocation.

void climatology (ctl t \*ctl, atm t \*atm)

Interpolate climatological data.

double ctmco2 (double nu, double p, double t, double u)

Compute carbon dioxide continuum (optical depth).

• double ctmh2o (double nu, double p, double t, double q, double u)

Compute water vapor continuum (optical depth).

```
    double ctmn2 (double nu, double p, double t)

      Compute nitrogen continuum (absorption coefficient).

    double ctmo2 (double nu, double p, double t)

      Compute oxygen continuum (absorption coefficient).

    void copy_atm (ctl_t *ctl, atm_t *atm_dest, atm_t *atm_src, int init)

      Copy and initialize atmospheric data.
• void copy_obs (ctl_t *ctl, obs_t *obs_dest, obs_t *obs_src, int init)
      Copy and initialize observation data.

    int find_emitter (ctl_t *ctl, const char *emitter)

      Find index of an emitter.

    void formod (ctl t *ctl, atm t *atm, obs t *obs)

      Determine ray paths and compute radiative transfer.

    void formod_continua (ctl_t *ctl, los_t *los, int ip, double *beta)

      Compute absorption coefficient of continua.

    void formod fov (ctl t *ctl, obs t *obs)

      Apply field of view convolution.

    void formod_pencil (ctl_t *ctl, atm_t *atm, obs_t *obs, int ir)

      Compute radiative transfer for a pencil beam.
• void formod_srcfunc (ctl_t *ctl, tbl_t *tbl, double t, double *src)
      Compute Planck source function.

    void geo2cart (double z, double lon, double lat, double *x)

      Convert geolocation to Cartesian coordinates.
• double gravity (double z, double lat)
      Determine gravity of Earth.

    void hydrostatic (ctl_t *ctl, atm_t *atm)

      Set hydrostatic equilibrium.

    void idx2name (ctl_t *ctl, int idx, char *quantity)

      Determine name of state vector quantity for given index.
void init_tbl (ctl_t *ctl, tbl_t *tbl)
      Initialize look-up tables.

    void intpol_atm (ctl_t *ctl, atm_t *atm, double z, double *p, double *t, double *q, double *k)

      Interpolate atmospheric data.
• void intpol_tbl (ctl_t *ctl, tbl_t *tbl, los_t *los, int ip, double tau_path[NG][ND], double tau_seg[ND])
      Get transmittance from look-up tables.
• double intpol_tbl_eps (tbl_t *tbl, int ig, int id, int ip, int it, double u)
      Interpolate emissivity from look-up tables.

    double intpol tbl u (tbl t *tbl, int ig, int id, int ip, int it, double eps)

      Interpolate column density from look-up tables.

    void jsec2time (double jsec, int *year, int *mon, int *day, int *hour, int *min, int *sec, double *remain)

      Convert seconds to date.

    void kernel (ctl t *ctl, atm t *atm, obs t *obs, gsl matrix *k)

      Compute Jacobians.
• int locate (double *xx, int n, double x)
      Find array index.
• int locate tbl (float *xx, int n, double x)
      Find array index in float array.

    size_t obs2y (ctl_t *ctl, obs_t *obs, gsl_vector *y, int *ida, int *ira)

      Compose measurement vector.

    double planck (double t, double nu)

      Compute Planck function.

    void raytrace (ctl_t *ctl, atm_t *atm, obs_t *obs, los_t *los, int ir)
```

Do ray-tracing to determine LOS. void read\_atm (const char \*dirname, const char \*filename, ctl\_t \*ctl, atm\_t \*atm) Read atmospheric data. void read\_ctl (int argc, char \*argv[], ctl\_t \*ctl) Read forward model control parameters. void read\_matrix (const char \*dirname, const char \*filename, gsl\_matrix \*matrix) Read matrix. void read\_obs (const char \*dirname, const char \*filename, ctl\_t \*ctl, obs\_t \*obs) Read observation data. void read shape (const char \*filename, double \*x, double \*y, int \*n) Read shape function. double refractivity (double p, double t) Compute refractivity (return value is n - 1). double scan\_ctl (int argc, char \*argv[], const char \*varname, int arridx, const char \*defvalue, char \*value) Search control parameter file for variable entry. void tangent\_point (los\_t \*los, double \*tpz, double \*tplon, double \*tplat) Find tangent point of a given LOS. void time2jsec (int year, int mon, int day, int hour, int min, int sec, double remain, double \*jsec) Convert date to seconds. void timer (const char \*name, const char \*file, const char \*func, int line, int mode) Measure wall-clock time. void write\_atm (const char \*dirname, const char \*filename, ctl\_t \*ctl, atm\_t \*atm) Write atmospheric data. void write matrix (const char \*dirname, const char \*filename, ctl t \*ctl, gsl matrix \*matrix, atm t \*atm, obs t \*obs, const char \*rowspace, const char \*colspace, const char \*sort) Write matrix. • void write\_obs (const char \*dirname, const char \*filename, ctl\_t \*ctl, obs\_t \*obs) Write observation data. void x2atm (ctl\_t \*ctl, gsl\_vector \*x, atm\_t \*atm) Decompose parameter vector or state vector. void x2atm\_help (atm\_t \*atm, double zmin, double zmax, double \*value, gsl\_vector \*x, size\_t \*n) Extract elements from state vector. void y2obs (ctl\_t \*ctl, gsl\_vector \*y, obs\_t \*obs) Decompose measurement vector. 5.13.1 Detailed Description JURASSIC library definitions. Definition in file jurassic.c.

# 5.13.2 Function Documentation

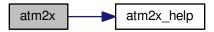
```
5.13.2.1 size_t atm2x ( ctl_t * ctl, atm_t * atm, gsl_vector * x, int * iqa, int * ipa )
```

Compose state vector or parameter vector.

Definition at line 29 of file jurassic.c.

```
00034
00035
00036
        int ig, iw;
00037
00038
        size t n = 0;
00039
00040
        /* Add pressure... */
00041
        atm2x_help(atm, ctl->retp_zmin, ctl->retp_zmax,
00042
                   atm->p, IDXP, x, iqa, ipa, &n);
00043
00044
        /* Add temperature... */
        atm2x_help(atm, ctl->rett_zmin, ctl->rett_zmax,
00045
00046
                    atm->t, IDXT, x, iqa, ipa, &n);
00047
00048
        /\star Add volume mixing ratios...
        for (ig = 0; ig < ctl->ng; ig++)
  atm2x_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
00049
00050
00051
                      atm->q[ig], IDXQ(ig), x, iqa, ipa, &n);
00052
00053
        /* Add extinction... */
00054
        for (iw = 0; iw < ctl->nw; iw++)
00055
          atm2x_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
                      atm->k[iw], IDXK(iw), x, iqa, ipa, &n);
00056
00057
00058
        return n;
00059 }
```

Here is the call graph for this function:



5.13.2.2 void atm2x\_help ( atm $_t * atm$ , double zmin, double zmax, double \* value, int  $val\_iqa$ ,  $gsl\_vector * x$ , int \* iqa, int \* ipa,  $size_t * n$ )

Add elements to state vector.

Definition at line 63 of file jurassic.c.

```
00072
00073
00074
        int ip;
00075
00076
        /* Add elements to state vector... */
00077
        for (ip = 0; ip < atm->np; ip++)
00078
         if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {
00079
            if (x != NULL)
08000
              gsl_vector_set(x, *n, value[ip]);
            if (iqa != NULL)
  iqa[*n] = val_iqa;
00081
00082
00083
            if (ipa != NULL)
00084
               ipa[*n] = ip;
00085
             (*n)++;
00086
          }
00087 }
```

5.13.2.3 double brightness ( double rad, double nu )

Compute brightness temperature.

Definition at line 91 of file jurassic.c.

```
00093 {
00094
00095    return C2 * nu / gsl_log1p(C1 * gsl_pow_3(nu) / rad);
00096 }
```

```
5.13.2.4 void cart2geo ( double * x, double * z, double * lon, double * lat )
```

Convert Cartesian coordinates to geolocation.

Definition at line 101 of file jurassic.c.

5.13.2.5 void climatology ( ctl\_t \* ctl, atm\_t \* atm\_mean )

Interpolate climatological data.

Definition at line 117 of file jurassic.c.

```
00119
00120
           static double z[121] = {
00121
              0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
              20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,
00123
00124
             56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107,
00125
00126
00127
             108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120
00128
00129
00130
           static double pre[121] = {
    1017, 901.083, 796.45, 702.227, 617.614, 541.644, 473.437, 412.288,
00131
00132
              357.603, 308.96, 265.994, 228.348, 195.619, 167.351, 143.039, 122.198, 104.369, 89.141, 76.1528, 65.0804, 55.641, 47.591, 40.7233, 34.8637,
00133
00134
00135
              29.8633, 25.5956, 21.9534, 18.8445, 16.1909, 13.9258, 11.9913,
00136
              10.34, 8.92988, 7.72454, 6.6924, 5.80701, 5.04654, 4.39238, 3.82902,
             3.34337, 2.92413, 2.56128, 2.2464, 1.97258, 1.73384, 1.52519, 1.34242, 1.18197, 1.04086, 0.916546, 0.806832, 0.709875, 0.624101, 0.548176,
00137
00138
             0.480974, 0.421507, 0.368904, 0.322408, 0.281386, 0.245249, 0.213465, 0.185549, 0.161072, 0.139644, 0.120913, 0.104568, 0.0903249, 0.0779269,
00139
00140
              0.0671493, 0.0577962, 0.0496902, 0.0426736, 0.0366093, 0.0313743,
00142
              0.0268598, 0.0229699, 0.0196206, 0.0167399, 0.0142646, 0.0121397,
00143
              0.0103181,\ 0.00875775,\ 0.00742226,\ 0.00628076,\ 0.00530519,\ 0.00447183,
             0.00376124, 0.00315632, 0.00264248, 0.00220738, 0.00184003, 0.00153095, 0.00127204, 0.00105608, 0.000876652, 0.00072798, 0.00060492,
00144
00145
00146
              0.000503201, 0.000419226, 0.000349896, 0.000292659, 0.000245421,
              0.000206394, 0.000174125, 0.000147441, 0.000125333, 0.000106985,
00147
              9.173e-05, 7.90172e-05, 6.84172e-05, 5.95574e-05, 5.21183e-05,
00148
              4.58348e-05, 4.05127e-05, 3.59987e-05, 3.21583e-05, 2.88718e-05,
00149
00150
             2.60322e-05, 2.35687e-05, 2.14263e-05, 1.95489e-05
00151
00152
00153
           static double tem[121]
                                            = {
00154
             285.14, 279.34, 273.91, 268.3, 263.24, 256.55, 250.2, 242.82, 236.17,
00155
              229.87, 225.04, 221.19, 218.85, 217.19, 216.2, 215.68, 215.42, 215.55,
             215.92, 216.4, 216.93, 217.45, 218.68, 219.39, 220.25, 221.3, 222.41, 223.88, 225.42, 227.2, 229.52, 231.89, 234.51, 236.85, 239.42, 241.94, 244.57, 247.36, 250.32, 253.34, 255.82, 258.27, 260.39, 262.03, 263.45, 264.2, 264.78, 264.67, 264.38, 263.24, 262.03, 260.02, 258.09, 255.63, 253.28, 250.43, 247.81, 245.26, 242.77, 240.38,
00156
00157
00158
00159
00160
00161
              237.94, 235.79, 233.53, 231.5, 229.53, 227.6, 225.62, 223.77, 222.06,
00162
              220.33, 218.69, 217.18, 215.64, 214.13, 212.52, 210.86, 209.25,
             207.49, 205.81, 204.11, 202.22, 200.32, 198.39, 195.92, 193.46, 190.94, 188.31, 185.82, 183.57, 181.43, 179.74, 178.64, 178.1, 178.25, 178.7, 179.41, 180.67, 182.31, 184.18, 186.6, 189.53, 192.66, 196.54,
00163
00164
00165
              201.13, 205.93, 211.73, 217.86, 225, 233.53, 242.57, 252.14, 261.48,
00166
00167
              272.97, 285.26, 299.12, 312.2, 324.17, 338.34, 352.56, 365.28
00168
00169
00170
           static double c2h2[121] = {
             1.352e-09, 2.83e-10, 1.269e-10, 6.926e-11, 4.346e-11, 2.909e-11,
              2.014e-11, 1.363e-11, 8.71e-12, 5.237e-12, 2.718e-12, 1.375e-12,
```

```
5.786e-13, 2.16e-13, 7.317e-14, 2.551e-14, 1.055e-14, 4.758e-15,
                2.056e-15, 7.703e-16, 2.82e-16, 1.035e-16, 4.382e-17, 1.946e-17, 9.638e-18, 5.2e-18, 2.811e-18, 1.494e-18, 7.925e-19, 4.213e-19,
00174
00175
00176
                1.998e-19, 8.78e-20, 3.877e-20, 1.728e-20, 7.743e-21, 3.536e-21,
                1.623e-21, 7.508e-22, 3.508e-22, 1.65e-22, 7.837e-23, 3.733e-23, 1.808e-23, 8.77e-24, 4.285e-24, 2.095e-24, 1.032e-24, 5.082e-25,
00177
00178
00179
                 2.506e-25, 1.236e-25, 6.088e-26, 2.996e-26, 1.465e-26, 0, 0, 0,
                00180
00181
                00182
                00183
00184
00185
             static double c2h6[121] =
                2.667e-09, 2.02e-09, 1.658e-09, 1.404e-09, 1.234e-09, 1.109e-09,
00186
00187
                1.012e-09, 9.262e-10, 8.472e-10, 7.71e-10, 6.932e-10, 6.216e-10,
00188
                 5.503e-10, 4.87e-10, 4.342e-10, 3.861e-10, 3.347e-10, 2.772e-10,
                2.209e-10, 1.672e-10, 1.197e-10, 8.536e-11, 5.783e-11, 3.846e-11,
00189
                2.495e-11, 1.592e-11, 1.017e-11, 6.327e-12, 3.895e-12, 2.403e-12,
00190
                 1.416e-12, 8.101e-13, 4.649e-13, 2.686e-13, 1.557e-13, 9.14e-14,
                 5.386e-14, 3.19e-14, 1.903e-14, 1.14e-14, 6.875e-15, 4.154e-15,
00192
                2.538e-15, 1.553e-15, 9.548e-16, 5.872e-16, 3.63e-16, 2.244e-16, 1.388e-16, 8.587e-17, 5.308e-17, 3.279e-17, 2.017e-17, 1.238e-17,
00193
00194
                7.542e-18, 4.585e-18, 2.776e-18, 1.671e-18, 9.985e-19, 5.937e-19, 3.518e-19, 2.07e-19, 1.215e-19, 7.06e-20, 4.097e-20, 2.37e-20, 1.363e-20, 7.802e-21, 4.441e-21, 2.523e-21, 1.424e-21, 8.015e-22,
00195
00196
00197
00198
                 4.497e-22, 2.505e-22, 1.391e-22, 7.691e-23, 4.238e-23, 2.331e-23,
                 1.274e-23, 6.929e-24, 3.752e-24, 2.02e-24, 1.083e-24, 5.774e-25,
00199
00200
                3.041e-25, 1.593e-25, 8.308e-26, 4.299e-26, 2.195e-26, 1.112e-26,
00201
                0, 0, 0, 0, 0, 0, 0, 0
00202
00203
00204
00205
             static double cc14[121] =
00206
                1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10,
                1.075e-10, 1.075e-10, 1.075e-10, 1.06e-10, 1.024e-10, 9.69e-11, 8.93e-11, 8.078e-11, 7.213e-11, 6.307e-11, 5.383e-11, 4.49e-11, 3.609e-11, 2.705e-11, 1.935e-11, 1.385e-11, 8.35e-12, 5.485e-12, 3.853e-12, 2.22e-12, 5.875e-13, 3.445e-13, 1.015e-13, 6.075e-14,
00207
00208
00209
00211
                 4.383e-14, 2.692e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00212
                 le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14,
00213
                1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00214
                1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
                le-14, le
00215
00216
00217
                 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00218
                 le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14,
00219
                1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00220
                1e-14, 1e-14, 1e-14
00221
00222
             static double ch4[121] = {
               1.864e-06, 1.835e-06, 1.819e-06, 1.805e-06, 1.796e-06, 1.788e-06,
00224
00225
                1.782e-06, 1.776e-06, 1.769e-06, 1.761e-06, 1.749e-06, 1.734e-06,
00226
                1.716e-06, 1.692e-06, 1.654e-06, 1.61e-06, 1.567e-06, 1.502e-06,
                1.433e-06, 1.371e-06, 1.323e-06, 1.277e-06, 1.232e-06, 1.188e-06, 1.147e-06, 1.108e-06, 1.07e-06, 1.027e-06, 9.854e-07, 9.416e-07, 8.933e-07, 8.478e-07, 7.988e-07, 7.515e-07, 7.07e-07, 6.64e-07,
00227
00228
00230
                6.239e-07, 5.864e-07, 5.512e-07, 5.184e-07, 4.87e-07, 4.571e-07,
                4.296e-07, 4.04e-07, 3.802e-07, 3.578e-07, 3.383e-07, 3.203e-07, 3.032e-07, 2.889e-07, 2.76e-07, 2.635e-07, 2.519e-07, 2.409e-07,
00231
00232
                2.302e-07, 2.219e-07, 2.144e-07, 2.071e-07, 1.999e-07, 1.93e-07, 1.862e-07, 1.795e-07, 1.731e-07, 1.668e-07, 1.607e-07, 1.548e-07,
00233
00234
                1.49e-07, 1.434e-07, 1.38e-07, 1.328e-07, 1.277e-07, 1.227e-07, 1.18e-07, 1.134e-07, 1.089e-07, 1.046e-07, 1.004e-07, 9.635e-08,
00236
00237
                 9.245e-08, 8.867e-08, 8.502e-08, 8.15e-08, 7.809e-08, 7.48e-08,
00238
                7.159e-08, 6.849e-08, 6.55e-08, 6.262e-08, 5.98e-08, 5.708e-08,
00239
                5.448e-08, 5.194e-08, 4.951e-08, 4.72e-08, 4.5e-08, 4.291e-08, 4.093e-08, 3.905e-08, 3.729e-08, 3.563e-08, 3.408e-08, 3.265e-08,
00240
00241
                3.128e-08, 2.996e-08, 2.87e-08, 2.76e-08, 2.657e-08, 2.558e-08,
                 2.467e-08, 2.385e-08, 2.307e-08, 2.234e-08, 2.168e-08, 2.108e-08,
00242
00243
                 2.05e-08, 1.998e-08, 1.947e-08, 1.902e-08, 1.86e-08, 1.819e-08,
                1.782e-08
00244
00245
             };
00246
00247
             static double clo[121] = {
                7.419e-15, 1.061e-14, 1.518e-14, 2.195e-14, 3.175e-14, 4.666e-14,
00248
00249
                 6.872e-14, 1.03e-13, 1.553e-13, 2.375e-13, 3.664e-13, 5.684e-13,
00250
                 8.915e-13, 1.402e-12, 2.269e-12, 4.125e-12, 7.501e-12, 1.257e-11,
                2.048e-11, 3.338e-11, 5.44e-11, 8.846e-11, 1.008e-10, 1.082e-10, 1.157e-10, 1.232e-10, 1.312e-10, 1.539e-10, 1.822e-10, 2.118e-10,
00251
00252
                2.387e-10, 2.687e-10, 2.875e-10, 3.031e-10, 3.23e-10, 3.648e-10, 4.117e-10, 4.477e-10, 4.633e-10, 4.794e-10, 4.95e-10, 5.104e-10,
00253
                 5.259e-10, 5.062e-10, 4.742e-10, 4.443e-10, 4.051e-10, 3.659e-10,
00255
00256
                3.305e-10, 2.911e-10, 2.54e-10, 2.215e-10, 1.927e-10, 1.675e-10,
00257
                1.452e-10, 1.259e-10, 1.09e-10, 9.416e-11, 8.119e-11, 6.991e-11,
                6.015e-11, 5.163e-11, 4.43e-11, 3.789e-11, 3.24e-11, 2.769e-11, 2.361e-11, 2.011e-11, 1.71e-11, 1.453e-11, 1.233e-11, 1.045e-11,
00258
00259
```

```
8.851e-12, 7.48e-12, 6.316e-12, 5.326e-12, 4.487e-12, 3.778e-12,
                3.176e-12, 2.665e-12, 2.234e-12, 1.87e-12, 1.563e-12, 1.304e-12, 1.085e-12, 9.007e-13, 7.468e-13, 6.179e-13, 5.092e-13, 4.188e-13,
00261
00262
00263
                3.442e-13, 2.816e-13, 2.304e-13, 1.885e-13, 1.542e-13, 1.263e-13,
                1.035e-13, 8.5e-14, 7.004e-14, 5.783e-14, 4.795e-14, 4.007e-14, 3.345e-14, 2.792e-14, 2.33e-14, 1.978e-14, 1.686e-14, 1.438e-14,
00264
00265
                1.234e-14, 1.07e-14, 9.312e-15, 8.131e-15, 7.164e-15, 6.367e-15,
                 5.67e-15, 5.088e-15, 4.565e-15, 4.138e-15, 3.769e-15, 3.432e-15,
00267
00268
                3.148e-15
00269
00270
             static double clono2[121] = {
00271
               1.011e-13, 1.515e-13, 2.272e-13, 3.446e-13, 5.231e-13, 8.085e-13,
00272
                1.253e-12, 1.979e-12, 3.149e-12, 5.092e-12, 8.312e-12, 1.366e-11,
00273
00274
                2.272e-11, 3.791e-11, 6.209e-11, 9.101e-11, 1.334e-10, 1.951e-10,
00275
                2.853e-10, 3.94e-10, 4.771e-10, 5.771e-10, 6.675e-10, 7.665e-10,
                8.504e-10, 8.924e-10, 9.363e-10, 8.923e-10, 8.411e-10, 7.646e-10,
00276
                6.525e-10, 5.576e-10, 4.398e-10, 3.403e-10, 2.612e-10, 1.915e-10,
00277
                1.407e-10, 1.028e-10, 7.455e-11, 5.42e-11, 3.708e-11, 2.438e-11,
                1.618e-11, 1.075e-11, 7.17e-12, 4.784e-12, 3.205e-12, 2.147e-12,
00279
                1.44e-12, 9.654e-13, 6.469e-13, 4.332e-13, 2.891e-13, 1.926e-13,
00280
00281
                1.274e-13, 8.422e-14, 5.547e-14, 3.636e-14, 2.368e-14, 1.536e-14,
                9.937e-15, 6.39e-15, 4.101e-15, 2.61e-15, 1.659e-15, 1.052e-15, 6.638e-16, 4.172e-16, 2.61e-16, 1.63e-16, 1.013e-16, 6.275e-17, 3.879e-17, 2.383e-17, 1.461e-17, 8.918e-18, 5.43e-18, 3.301e-18,
00282
00283
00284
                1.997e-18, 1.203e-18, 7.216e-19, 4.311e-19, 2.564e-19, 1.519e-19,
                8.911e-20, 5.203e-20, 3.026e-20, 1.748e-20, 9.99e-21, 5.673e-21,
00286
00287
                3.215e-21, 1.799e-21, 1.006e-21, 5.628e-22, 3.146e-22, 1.766e-22,
                9.94e-23, 5.614e-23, 3.206e-23, 1.841e-23, 1.071e-23, 6.366e-24,
00288
                3.776e-24, 2.238e-24, 1.326e-24, 8.253e-25, 5.201e-25, 3.279e-25, 2.108e-25, 1.395e-25, 9.326e-26, 6.299e-26, 4.365e-26, 3.104e-26,
00289
00290
00291
                2.219e-26, 1.621e-26, 1.185e-26, 8.92e-27, 6.804e-27, 5.191e-27,
00292
                4.041e-27
00293
00294
00295
             static double co[121] = {
               1.907e-07, 1.553e-07, 1.362e-07, 1.216e-07, 1.114e-07, 1.036e-07, 9.737e-08, 9.152e-08, 8.559e-08, 7.966e-08, 7.277e-08, 6.615e-08,
00296
00298
                 5.884e-08, 5.22e-08, 4.699e-08, 4.284e-08, 3.776e-08, 3.274e-08,
00299
                2.845e-08, 2.479e-08, 2.246e-08, 2.054e-08, 1.991e-08, 1.951e-08,
00300
                1.94e-08, 2.009e-08, 2.1e-08, 2.201e-08, 2.322e-08, 2.45e-08,
                2.602e-08, 2.73e-08, 2.867e-08, 2.998e-08, 3.135e-08, 3.255e-08,
00301
00302
                3.352e-08, 3.426e-08, 3.484e-08, 3.53e-08, 3.593e-08, 3.671e-08,
                3.759e-08, 3.945e-08, 4.192e-08, 4.49e-08, 5.03e-08, 5.703e-08,
00303
                6.538e-08, 7.878e-08, 9.644e-08, 1.196e-07, 1.498e-07, 1.904e-07,
00305
                2.422e-07, 3.055e-07, 3.804e-07, 4.747e-07, 5.899e-07,
                8.91e-07, 1.071e-06, 1.296e-06, 1.546e-06, 1.823e-06, 2.135e-06, 2.44e-06, 2.714e-06, 2.967e-06, 3.189e-06, 3.391e-06, 3.58e-06,
00306
00307
                3.773e-06, 4.022e-06, 4.346e-06, 4.749e-06, 5.199e-06, 5.668e-06, 6.157e-06, 6.688e-06, 7.254e-06, 7.867e-06, 8.539e-06, 9.26e-06,
00308
00309
                1.009e-05, 1.119e-05, 1.228e-05, 1.365e-05, 1.506e-05, 1.641e-05,
                 1.784e-05, 1.952e-05, 2.132e-05, 2.323e-05, 2.531e-05, 2.754e-05,
00311
00312
                3.047e-05, 3.459e-05, 3.922e-05, 4.439e-05, 4.825e-05, 5.077e-05,
00313
                5.34 e-05, \ 5.618 e-05, \ 5.909 e-05, \ 6.207 e-05, \ 6.519 e-05, \ 6.845 e-05,
                6.819e-05, 6.726e-05, 6.622e-05, 6.512e-05, 6.671e-05, 6.862e-05, 7.048e-05, 7.264e-05, 7.3e-05, 7.2e-05, 7.2e-
00314
00315
00317
00318
             static double cof2[121] = {
00319
                7.5e-14, 1.055e-13, 1.485e-13, 2.111e-13, 3.001e-13, 4.333e-13,
                6.269e-13, 9.221e-13, 1.364e-12, 2.046e-12, 3.093e-12, 4.703e-12, 7.225e-12, 1.113e-11, 1.66e-11, 2.088e-11, 2.626e-11, 3.433e-11, 4.549e-11, 5.886e-11, 7.21e-11, 8.824e-11, 1.015e-10, 1.155e-10,
00320
00321
00322
                1.288e-10, 1.388e-10, 1.497e-10, 1.554e-10, 1.606e-10, 1.639e-10,
00323
00324
                1.64e-10, 1.64e-10, 1.596e-10, 1.542e-10, 1.482e-10, 1.382e-10,
00325
                1.289e-10, 1.198e-10, 1.109e-10, 1.026e-10, 9.484e-11, 8.75e-11,
                8.086e-11, 7.49e-11, 6.948e-11, 6.446e-11, 5.961e-11, 5.505e-11, 5.085e-11, 4.586e-11, 4.1e-11, 3.665e-11, 3.235e-11, 2.842e-11, 2.491e-11, 2.11e-11, 1.769e-11, 1.479e-11, 1.197e-11, 9.631e-12,
00326
00327
00328
                7.74e-12, 6.201e-12, 4.963e-12, 3.956e-12, 3.151e-12, 2.507e-12, 1.99e-12, 1.576e-12, 1.245e-12, 9.83e-13, 7.742e-13, 6.088e-13,
00330
00331
                4.782e-13, 3.745e-13, 2.929e-13, 2.286e-13, 1.782e-13, 1.388e-13,
00332
                1.079e-13, 8.362e-14, 6.471e-14, 4.996e-14, 3.85e-14, 2.96e-14,
                2.265e-14, 1.729e-14, 1.317e-14, 9.998e-15, 7.549e-15, 5.683e-15,
00333
                 4.273e-15, 3.193e-15, 2.385e-15, 1.782e-15, 1.331e-15, 9.957e-16,
00334
                7.461e-16, 5.601e-16, 4.228e-16, 3.201e-16, 2.438e-16, 1.878e-16,
00335
                1.445e-16, 1.111e-16, 8.544e-17, 6.734e-17, 5.341e-17, 4.237e-17,
00336
00337
                3.394e-17, 2.759e-17, 2.254e-17, 1.851e-17, 1.54e-17, 1.297e-17,
                1.096e-17, 9.365e-18, 8e-18, 6.938e-18, 6.056e-18, 5.287e-18,
00338
00339
                4.662e-18
00340
00341
00342
             static double f11[121] =
00343
                2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10,
00344
                2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.635e-10, 2.536e-10,
                2.44e-10, 2.348e-10, 2.258e-10, 2.153e-10, 2.046e-10, 1.929e-10, 1.782e-10, 1.648e-10, 1.463e-10, 1.291e-10, 1.1e-10, 8.874e-11,
00345
00346
```

```
00347
               7.165e-11, 5.201e-11, 3.744e-11, 2.577e-11, 1.64e-11, 1.048e-11,
               5.993e-12, 3.345e-12, 1.839e-12, 9.264e-13, 4.688e-13, 2.329e-13,
00348
00349
               1.129e-13, 5.505e-14, 2.825e-14, 1.492e-14, 7.997e-15, 5.384e-15,
00350
               3.988e-15, 2.955e-15, 2.196e-15, 1.632e-15, 1.214e-15, 9.025e-16,
00351
               6.708e-16, 4.984e-16, 3.693e-16, 2.733e-16, 2.013e-16, 1.481e-16,
               1.087e-16, 7.945e-17, 5.782e-17, 4.195e-17, 3.038e-17, 2.19e-17,
00352
               1.577e-17, 1.128e-17, 8.063e-18, 5.753e-18, 4.09e-18, 2.899e-18,
               2.048e-18, 1.444e-18, 1.015e-18,
                                                                  7.12e-19, 4.985e-19, 3.474e-19,
00354
00355
               2.417e-19, 1.677e-19, 1.161e-19, 8.029e-20, 5.533e-20, 3.799e-20,
               2.602e-20, 1.776e-20, 1.209e-20, 8.202e-21, 5.522e-21, 3.707e-21, 2.48e-21, 1.652e-21, 1.091e-21, 7.174e-22, 4.709e-22, 3.063e-22, 1.991e-22, 1.294e-22, 8.412e-23, 5.483e-23, 3.581e-23, 2.345e-23,
00356
00357
00358
00359
               1.548e-23, 1.027e-23, 6.869e-24, 4.673e-24, 3.173e-24, 2.153e-24,
               1.461e-24, 1.028e-24, 7.302e-25, 5.188e-25, 3.739e-25, 2.753e-25,
00360
00361
               2.043e-25, 1.528e-25, 1.164e-25, 9.041e-26, 7.051e-26, 5.587e-26,
00362
               4.428e-26, 3.588e-26, 2.936e-26, 2.402e-26, 1.995e-26
00363
00364
           static double f12[121] = {
              5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10,
00366
               5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.429e-10, 5.291e-10,
00367
00368
               5.155e-10, 5.022e-10, 4.893e-10, 4.772e-10, 4.655e-10, 4.497e-10,
00369
               4.249e-10, 4.015e-10, 3.632e-10, 3.261e-10, 2.858e-10, 2.408e-10,
00370
               2.03e-10, 1.685e-10, 1.4e-10, 1.163e-10, 9.65e-11, 8.02e-11, 6.705e-11,
00371
               5.624e-11, 4.764e-11, 4.249e-11, 3.792e-11, 3.315e-11, 2.819e-11,
               2.4e-11, 1.999e-11, 1.64e-11, 1.352e-11, 1.14e-11, 9.714e-12,
00372
00373
               8.28e-12, 7.176e-12, 6.251e-12, 5.446e-12, 4.72e-12, 4.081e-12,
00374
               3.528e-12, 3.08e-12, 2.699e-12, 2.359e-12, 2.111e-12, 1.901e-12,
00375
               1.709e-12, 1.534e-12, 1.376e-12, 1.233e-12, 1.103e-12, 9.869e-13,
00376
               8.808e-13, 7.859e-13, 7.008e-13, 6.241e-13, 5.553e-13, 4.935e-13,
00377
               4.383e-13, 3.889e-13, 3.447e-13, 3.054e-13, 2.702e-13, 2.389e-13,
00378
               2.11e-13, 1.862e-13, 1.643e-13, 1.448e-13, 1.274e-13, 1.121e-13,
               9.844e-14, 8.638e-14, 7.572e-14, 6.62e-14, 5.782e-14, 5.045e-14,
00379
00380
               4.394e-14, 3.817e-14, 3.311e-14, 2.87e-14, 2.48e-14, 2.142e-14,
00381
               1.851e-14, 1.599e-14, 1.383e-14, 1.196e-14, 1.036e-14, 9e-15,
               7.828e-15, 6.829e-15, 5.992e-15, 5.254e-15, 4.606e-15, 4.037e-15,
00382
               3.583e-15, 3.19e-15, 2.841e-15, 2.542e-15, 2.291e-15, 2.07e-15, 1.875e-15, 1.71e-15, 1.57e-15, 1.442e-15, 1.333e-15, 1.232e-15,
00383
00385
               1.147e-15, 1.071e-15, 1.001e-15, 9.396e-16
00386
00387
00388
           static double f14[121] = {
              9e-11, 8.73e-11, 8.46e-11,
00389
00390
               8.19e-11, 7.92e-11, 7.74e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00391
00392
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00393
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00394
               7.65e-11,\ 7.65e-11,\ 7.65e-11,\ 7.65e-11,\ 7.65e-11,\ 7.65e-11,
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00395
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00396
                                                                                                           7.65e-11.
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00398
                                                                                                           7.65e-11,
00399
               7.65e-11, 7.65e-11, 7.65e-11,
                                                             7.65e-11, 7.65e-11, 7.65e-11,
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-
00400
                                                                                                           7.65e-11,
00401
                                                                                                           7.65e-11,
00402
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00403
00404
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00405
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11
00406
00407
00408
           static double f22[121] = {
              1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10,
               1.4e-10, 1.4e-10, 1.4e-10, 1.372e-10, 1.317e-10, 1.235e-10, 1.153e-10,
00410
00411
               1.075e-10, 1.002e-10, 9.332e-11, 8.738e-11, 8.194e-11, 7.7e-11,
00412
               7.165e-11, 6.753e-11, 6.341e-11, 5.971e-11, 5.6e-11, 5.229e-11,
              4.859e-11, 4.488e-11, 4.118e-11, 3.83e-11, 3.568e-11, 3.308e-11, 3.047e-11, 2.82e-11, 2.594e-11, 2.409e-11, 2.237e-11, 2.065e-11, 1.894e-11, 1.771e-11, 1.647e-11, 1.532e-11, 1.416e-11, 1.332e-11,
00413
00414
00415
               1.246e-11, 1.161e-11, 1.087e-11, 1.017e-11, 9.471e-12, 8.853e-12,
               8.235e-12, 7.741e-12, 7.247e-12, 6.836e-12, 6.506e-12,
00417
                                                                                                  6.176e-12.
00418
               5.913e-12, 5.65e-12, 5.419e-12, 5.221e-12, 5.024e-12, 4.859e-12,
00419
               4.694e-12, 4.546e-12, 4.414e-12, 4.282e-12, 4.15e-12, 4.019e-12,
               3.903e-12, 3.805e-12, 3.706e-12, 3.607e-12, 3.508e-12, 3.41e-12, 3.31e-12, 3.212e-12, 3.129e-12, 3.047e-12, 2.964e-12, 2.882e-12, 2.8e-12, 2.734e-12, 2.668e-12, 2.602e-12, 2.537e-12, 2.471e-12,
00420
00421
00422
               2.421e-12, 2.372e-12, 2.322e-12, 2.273e-12, 2.224e-12, 2.182e-12,
00423
00424
               2.141e-12, 2.1e-12, 2.059e-12, 2.018e-12, 1.977e-12, 1.935e-12,
00425
               1.894e-12, 1.853e-12, 1.812e-12, 1.77e-12, 1.73e-12, 1.688e-12,
               1.647e-12, 1.606e-12, 1.565e-12, 1.524e-12, 1.483e-12, 1.441e-12, 1.4e-12, 1.359e-12, 1.317e-12, 1.276e-12, 1.235e-12, 1.194e-12,
00426
00427
               1.153e-12, 1.112e-12, 1.071e-12, 1.029e-12, 9.883e-13
00428
00429
00430
00431
           static double h2o[121] = {
              0.01166, 0.008269, 0.005742, 0.003845, 0.00277, 0.001897, 0.001272,
00432
00433
              0.000827, 0.000539, 0.0003469, 0.0001579, 3.134e-05, 1.341e-05,
```

```
6.764e-06, 4.498e-06, 3.703e-06, 3.724e-06, 3.899e-06, 4.002e-06,
           4.122e-06, 4.277e-06, 4.438e-06, 4.558e-06, 4.673e-06, 4.763e-06,
00435
00436
           4.809e-06, 4.856e-06, 4.936e-06, 5.021e-06, 5.114e-06, 5.222e-06,
00437
           5.331e-06, 5.414e-06, 5.488e-06, 5.563e-06, 5.633e-06, 5.704e-06,
00438
           5.767e-06, 5.819e-06, 5.872e-06, 5.914e-06, 5.949e-06, 5.984e-06,
           6.015e-06, 6.044e-06, 6.073e-06, 6.104e-06, 6.136e-06, 6.167e-06,
00439
           6.189e-06, 6.208e-06, 6.226e-06, 6.212e-06, 6.185e-06, 6.158e-06,
           6.114e-06, 6.066e-06, 6.018e-06, 5.877e-06, 5.728e-06, 5.582e-06,
00441
00442
           5.437e-06, 5.296e-06, 5.156e-06, 5.02e-06, 4.886e-06, 4.754e-06,
00443
           4.625e-06, 4.498e-06, 4.374e-06, 4.242e-06, 4.096e-06, 3.955e-06,
           3.817e-06, 3.683e-06, 3.491e-06, 3.204e-06, 2.94e-06, 2.696e-06,
00444
           2.47e-06, 2.252e-06, 2.019e-06, 1.808e-06, 1.618e-06, 1.445e-06,
00445
           1.285e-06, 1.105e-06, 9.489e-07, 8.121e-07, 6.938e-07, 5.924e-07,
00446
           5.04e-07, 4.288e-07, 3.648e-07, 3.103e-07, 2.642e-07, 2.252e-07,
00447
00448
           1.921e-07, 1.643e-07, 1.408e-07, 1.211e-07, 1.048e-07, 9.063e-08,
           7.835e-08, 6.774e-08, 5.936e-08, 5.221e-08, 4.592e-08, 4.061e-08, 3.62e-08, 3.236e-08, 2.902e-08, 2.62e-08, 2.383e-08, 2.171e-08,
00449
00450
00451
           1.989e-08, 1.823e-08, 1.684e-08, 1.562e-08, 1.449e-08, 1.351e-08
00452
00453
00454
         static double h2o2[121] =
00455
           1.779e-10, 7.938e-10, 8.953e-10, 8.032e-10, 6.564e-10, 5.159e-10,
00456
           4.003e-10, 3.026e-10, 2.222e-10, 1.58e-10, 1.044e-10, 6.605e-11,
          3.413e-11, 1.453e-11, 1.062e-11, 1.009e-11, 9.597e-12, 1.175e-11, 1.572e-11, 2.091e-11, 2.746e-11, 3.603e-11, 4.791e-11, 6.387e-11,
00457
00458
           8.239e-11, 1.007e-10, 1.23e-10, 1.363e-10, 1.489e-10, 1.585e-10,
           1.608e-10, 1.632e-10, 1.576e-10, 1.502e-10, 1.423e-10, 1.302e-10,
00460
00461
           1.192e-10, 1.085e-10, 9.795e-11, 8.854e-11, 8.057e-11, 7.36e-11,
          6.736e-11, 6.362e-11, 6.087e-11, 5.825e-11, 5.623e-11, 5.443e-11, 5.27e-11, 5.098e-11, 4.931e-11, 4.769e-11, 4.611e-11, 4.458e-11, 4.308e-11, 4.102e-11, 3.887e-11, 3.682e-11, 3.521e-11, 3.369e-11,
00462
00463
00464
00465
           3.224e-11, 3.082e-11, 2.946e-11, 2.814e-11, 2.687e-11, 2.566e-11,
           2.449e-11, 2.336e-11, 2.227e-11, 2.123e-11, 2.023e-11, 1.927e-11,
00466
00467
           1.835e-11, 1.746e-11, 1.661e-11, 1.58e-11, 1.502e-11, 1.428e-11,
00468
           1.357e-11, 1.289e-11, 1.224e-11, 1.161e-11, 1.102e-11, 1.045e-11,
           9.895e-12, 9.369e-12, 8.866e-12, 8.386e-12, 7.922e-12, 7.479e-12,
00469
           7.06e-12, 6.656e-12, 6.274e-12, 5.914e-12, 5.575e-12, 5.257e-12, 4.959e-12, 4.679e-12, 4.42e-12, 4.178e-12, 3.954e-12, 3.75e-12,
00470
00472
           3.557e-12, 3.372e-12, 3.198e-12, 3.047e-12, 2.908e-12, 2.775e-12,
00473
           2.653e-12, 2.544e-12, 2.442e-12, 2.346e-12, 2.26e-12, 2.183e-12,
00474
           2.11e-12, 2.044e-12, 1.98e-12, 1.924e-12, 1.871e-12, 1.821e-12,
00475
           1.775e-12
00476
00477
00478
        static double hcn[121] = {
00479
           5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10,
00480
           5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.498e-10, 5.495e-10, 5.493e-10,
           5.49e-10, 5.488e-10, 4.717e-10, 3.946e-10, 3.174e-10, 2.4e-10,
00481
           1.626e-10, 1.619e-10, 1.612e-10, 1.602e-10, 1.593e-10, 1.582e-10,
00482
           1.572e-10, 1.56e-10, 1.549e-10, 1.539e-10, 1.53e-10, 1.519e-10,
00483
           1.506e-10, 1.487e-10, 1.467e-10, 1.449e-10, 1.43e-10, 1.413e-10,
           1.397e-10, 1.382e-10, 1.368e-10, 1.354e-10, 1.337e-10, 1.315e-10,
00485
00486
           1.292e-10, 1.267e-10, 1.241e-10, 1.215e-10, 1.19e-10, 1.165e-10,
00487
           1.141e-10, 1.118e-10, 1.096e-10, 1.072e-10, 1.047e-10, 1.021e-10,
           9.968e-11, 9.739e-11, 9.539e-11, 9.339e-11, 9.135e-11, 8.898e-11,
00488
           8.664e-11, 8.439e-11, 8.249e-11, 8.075e-11, 7.904e-11, 7.735e-11,
00489
           7.565e-11, 7.399e-11, 7.245e-11, 7.109e-11, 6.982e-11, 6.863e-11,
           6.755e-11, 6.657e-11, 6.587e-11, 6.527e-11, 6.476e-11, 6.428e-11,
00491
00492
           6.382e-11, 6.343e-11, 6.307e-11, 6.272e-11, 6.238e-11, 6.205e-11,
00493
           6.17e-11, 6.137e-11, 6.102e-11, 6.072e-11, 6.046e-11, 6.03e-11,
00494
           6.018e-11, 6.01e-11, 6.001e-11, 5.992e-11, 5.984e-11, 5.975e-11,
           5.967e-11, 5.958e-11, 5.95e-11, 5.941e-11, 5.933e-11, 5.925e-11, 5.916e-11, 5.908e-11, 5.899e-11, 5.891e-11, 5.883e-11, 5.874e-11,
00495
00496
           5.866e-11, 5.858e-11, 5.85e-11, 5.841e-11, 5.833e-11, 5.825e-11, 5.817e-11, 5.808e-11, 5.8e-11, 5.792e-11, 5.784e-11
00497
00498
00499
00500
00501
        static double hno3[121] = {
          1.809e-10, 7.234e-10, 5.899e-10, 4.342e-10, 3.277e-10, 2.661e-10,
00502
           2.35e-10, 2.267e-10, 2.389e-10, 2.651e-10, 3.255e-10, 4.099e-10,
           5.42e-10, 6.978e-10, 8.807e-10, 1.112e-09, 1.405e-09, 2.04e-09, 3.111e-09, 4.5e-09, 5.762e-09, 7.37e-09, 7.852e-09, 8.109e-09,
00504
00505
           8.067e-09, 7.554e-09, 7.076e-09, 6.268e-09, 5.524e-09, 4.749e-09, 3.909e-09, 3.223e-09, 2.517e-09, 1.942e-09, 1.493e-09, 1.122e-09,
00506
00507
           8.449e-10, 6.361e-10, 4.787e-10, 3.611e-10, 2.804e-10, 2.215e-10,
00508
           1.758e-10, 1.441e-10, 1.197e-10, 9.953e-11, 8.505e-11, 7.334e-11,
00509
           6.325e-11, 5.625e-11, 5.058e-11, 4.548e-11, 4.122e-11, 3.748e-11,
00510
00511
           3.402e-11, 3.088e-11, 2.8e-11, 2.536e-11, 2.293e-11, 2.072e-11,
00512
           1.871e-11, 1.687e-11, 1.52e-11, 1.368e-11, 1.23e-11, 1.105e-11,
           9.922e-12, 8.898e-12, 7.972e-12, 7.139e-12, 6.385e-12, 5.708e-12,
00513
           5.099e-12, 4.549e-12, 4.056e-12, 3.613e-12, 3.216e-12, 2.862e-12,
00514
           2.544e-12, 2.259e-12, 2.004e-12, 1.776e-12, 1.572e-12, 1.391e-12,
           1.227e-12, 1.082e-12, 9.528e-13, 8.379e-13, 7.349e-13, 6.436e-13,
00516
00517
           5.634e-13, 4.917e-13, 4.291e-13, 3.745e-13, 3.267e-13, 2.854e-13,
00518
           2.494e-13, 2.181e-13, 1.913e-13, 1.68e-13, 1.479e-13, 1.31e-13,
           1.159e-13, 1.025e-13, 9.067e-14, 8.113e-14, 7.281e-14, 6.535e-14, 5.892e-14, 5.348e-14, 4.867e-14, 4.439e-14, 4.073e-14, 3.76e-14,
00519
00520
```

```
3.476e-14, 3.229e-14, 3e-14, 2.807e-14, 2.635e-14, 2.473e-14,
00522
00523
00524
00525
            static double hno4[121] = {
               6.118e-12, 3.594e-12, 2.807e-12, 3.04e-12, 4.458e-12, 7.986e-12,
00526
               3.347e-11, 3.005e-11, 3.173e-11, 4.055e-11, 5.812e-11, 8.489e-11,
00528
00529
               1.19e-10, 1.482e-10, 1.766e-10, 2.103e-10, 2.35e-10, 2.598e-10,
00530
               2.801e-10, 2.899e-10, 3e-10, 2.817e-10, 2.617e-10, 2.332e-10,
               1.933e-10, 1.605e-10, 1.232e-10, 9.285e-11, 6.941e-11, 4.951e-11,
00531
               3.539e-11, 2.402e-11, 1.522e-11, 9.676e-12, 6.056e-12, 3.745e-12, 2.34e-12, 1.463e-12, 9.186e-13, 5.769e-13, 3.322e-13, 1.853e-13,
00532
00533
               1.035e-13, 7.173e-14, 5.382e-14, 4.036e-14, 3.401e-14, 2.997e-14,
00534
00535
               2.635e-14, 2.316e-14, 2.034e-14, 1.783e-14, 1.56e-14, 1.363e-14,
00536
               1.19e-14, 1.037e-14, 9.032e-15, 7.846e-15, 6.813e-15, 5.912e-15, 5.121e-15, 4.431e-15, 3.829e-15, 3.306e-15, 2.851e-15, 2.456e-15,
00537
00538
               2.114e-15, 1.816e-15, 1.559e-15, 1.337e-15, 1.146e-15, 9.811e-16,
               8.389e-16, 7.162e-16, 6.109e-16, 5.203e-16, 4.425e-16, 3.76e-16,
               3.184e-16, 2.692e-16, 2.274e-16, 1.917e-16, 1.61e-16, 1.35e-16,
00540
               1.131e-16, 9.437e-17, 7.874e-17, 6.57e-17, 5.481e-17, 4.579e-17, 3.828e-17, 3.204e-17, 2.691e-17, 2.264e-17, 1.912e-17, 1.626e-17,
00541
00542
00543
               1.382e-17, 1.174e-17, 9.972e-18, 8.603e-18, 7.45e-18, 6.453e-18,
               5.623e-18, 4.944e-18, 4.361e-18, 3.859e-18, 3.443e-18, 3.096e-18, 2.788e-18, 2.528e-18, 2.293e-18, 2.099e-18, 1.929e-18, 1.773e-18,
00544
00545
00546
               1.64e-18
00547
00548
00549
            static double hocl[121] = {
               1.056e-12, 1.194e-12, 1.35e-12, 1.531e-12, 1.737e-12, 1.982e-12,
00550
               2.263e-12, 2.599e-12, 2.991e-12, 3.459e-12, 4.012e-12, 4.662e-12, 5.438e-12, 6.35e-12, 7.425e-12, 8.686e-12, 1.016e-11, 1.188e-11,
00551
00553
               1.389e-11, 1.659e-11, 2.087e-11, 2.621e-11, 3.265e-11, 4.064e-11,
00554
                4.859e-11, 5.441e-11, 6.09e-11, 6.373e-11, 6.611e-11, 6.94e-11,
               7.44e-11, 7.97e-11, 8.775e-11, 9.722e-11, 1.064e-10, 1.089e-10, 1.114e-10, 1.106e-10, 1.053e-10, 1.004e-10, 9.006e-11, 7.778e-11,
00555
00556
00557
               6.739e-11, 5.636e-11, 4.655e-11, 3.845e-11, 3.042e-11, 2.368e-11,
               1.845e-11, 1.442e-11, 1.127e-11, 8.814e-12, 6.544e-12, 4.763e-12,
00559
                3.449e-12, 2.612e-12, 1.999e-12, 1.526e-12, 1.16e-12, 8.793e-13,
00560
                6.655e-13, 5.017e-13, 3.778e-13, 2.829e-13, 2.117e-13, 1.582e-13,
00561
               1.178e-13, 8.755e-14, 6.486e-14, 4.799e-14, 3.54e-14, 2.606e-14,
               1.916e-14, 1.403e-14, 1.026e-14, 7.48e-15, 5.446e-15, 3.961e-15,
00562
               2.872e-15, 2.076e-15, 1.498e-15, 1.077e-15, 7.726e-16, 5.528e-16,
00563
00564
               3.929e-16, 2.785e-16, 1.969e-16, 1.386e-16, 9.69e-17, 6.747e-17,
00565
                4.692e-17, 3.236e-17, 2.232e-17, 1.539e-17, 1.061e-17, 7.332e-18,
00566
               5.076e-18, 3.522e-18, 2.461e-18, 1.726e-18, 1.22e-18, 8.75e-19,
               6.264e-19, 4.482e-19, 3.207e-19, 2.368e-19, 1.762e-19, 1.312e-19, 9.891e-20, 7.595e-20, 5.87e-20, 4.567e-20, 3.612e-20, 2.904e-20, 2.343e-20, 1.917e-20, 1.568e-20, 1.308e-20, 1.1e-20, 9.25e-21,
00567
00568
00569
00570
                7.881e-21
00571
00572
00573
            static double n2o[121] = {
              3.17e-07, 3.03e-07,
00574
00575
00576
                2.984e-07, 2.938e-07, 2.892e-07, 2.847e-07, 2.779e-07, 2.705e-07,
                2.631e-07, 2.557e-07, 2.484e-07, 2.345e-07, 2.201e-07, 2.01e-07,
00577
                1.754e-07, 1.532e-07, 1.329e-07, 1.154e-07, 1.003e-07, 8.735e-08,
00578
00579
               7.617e-08, 6.512e-08, 5.547e-08, 4.709e-08, 3.915e-08, 3.259e-08,
00580
               2.738e-08, 2.327e-08, 1.98e-08, 1.711e-08, 1.493e-08, 1.306e-08,
00581
               1.165e-08, 1.049e-08, 9.439e-09, 8.375e-09, 7.391e-09, 6.525e-09,
00582
               5.759e-09, 5.083e-09, 4.485e-09, 3.953e-09, 3.601e-09, 3.27e-09,
               2.975e-09, 2.757e-09, 2.556e-09, 2.37e-09, 2.195e-09, 2.032e-09,
               1.912e-09, 1.79e-09, 1.679e-09, 1.572e-09, 1.482e-09, 1.402e-09,
00584
00585
               1.326e-09, 1.254e-09, 1.187e-09, 1.127e-09, 1.071e-09, 1.02e-09,
00586
               9.673e-10, 9.193e-10, 8.752e-10, 8.379e-10, 8.017e-10, 7.66e-10,
00587
               7.319e-10, 7.004e-10, 6.721e-10, 6.459e-10, 6.199e-10, 5.942e-10, 5.703e-10, 5.488e-10, 5.283e-10, 5.082e-10, 4.877e-10, 4.696e-10,
00588
00589
               4.52e-10, 4.355e-10, 4.198e-10, 4.039e-10, 3.888e-10, 3.754e-10,
                3.624e-10, 3.499e-10, 3.381e-10, 3.267e-10, 3.163e-10, 3.058e-10,
00591
                2.959e-10, 2.864e-10, 2.77e-10, 2.686e-10, 2.604e-10, 2.534e-10,
00592
               2.462e-10, 2.386e-10, 2.318e-10, 2.247e-10, 2.189e-10, 2.133e-10,
00593
               2.071e-10, 2.014e-10, 1.955e-10, 1.908e-10, 1.86e-10, 1.817e-10
00594
00595
00596
            static double n2o5[121] = {
               1.231e-11, 3.035e-12, 1.702e-12, 9.877e-13, 8.081e-13, 9.039e-13,
00597
00598
               1.169e-12, 1.474e-12, 1.651e-12, 1.795e-12, 1.998e-12, 2.543e-12,
                4.398e-12, 7.698e-12, 1.28e-11, 2.131e-11, 3.548e-11, 5.894e-11,
00599
               7.645e-11, 1.089e-10, 1.391e-10, 1.886e-10, 2.386e-10, 2.986e-10,
00600
               3.487e-10, 3.994e-10, 4.5e-10, 4.6e-10, 4.591e-10, 4.1e-10, 3.488e-10,
00601
                2.846e-10, 2.287e-10, 1.696e-10, 1.011e-10, 6.428e-11, 4.324e-11,
                2.225e-11, 6.214e-12, 3.608e-12, 8.793e-13, 4.491e-13, 1.04e-13,
00603
00604
                6.1e-14, 3.436e-14, 6.671e-15, 1.171e-15, 5.848e-16, 1.212e-16,
00605
               le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
               le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le
00606
00607
```

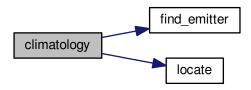
```
1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
                          le-16, le
00609
00610
00611
                          1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00612
                          1e-16, 1e-16
00613
00615
                     static double nh3[121] = {
00616
                        le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10,
00617
                           1e-10, 1e-10, 1e-10, 1e-10, 9.444e-11, 8.488e-11, 7.241e-11, 5.785e-11,
                          4.178e-11, 3.018e-11, 2.18e-11, 1.574e-11, 1.137e-11, 8.211e-12, 5.973e-12, 4.327e-12, 3.118e-12, 2.234e-12, 1.573e-12, 1.04e-12,
00618
00619
                           6.762e-13, 4.202e-13, 2.406e-13, 1.335e-13, 6.938e-14, 3.105e-14,
00620
                          1.609e-14, 1.033e-14, 6.432e-15, 4.031e-15, 2.555e-15, 1.656e-15,
00621
00622
                           1.115e-15, 7.904e-16, 5.63e-16, 4.048e-16, 2.876e-16, 2.004e-16,
                          1.356e-16, 9.237e-17, 6.235e-17, 4.223e-17, 3.009e-17, 2.328e-17, 2.002e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00623
00624
                           1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00625
                           1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                           1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                          1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 
00628
00629
00630
                          1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                          1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 
00631
00632
                           1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                           1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00634
00635
                          1.914e-17
00636
00637
00638
                     static double no[121] = {
00639
                          2.586e-10, 4.143e-11, 1.566e-11, 9.591e-12, 8.088e-12, 8.462e-12,
                           1.013e-11, 1.328e-11, 1.855e-11, 2.678e-11, 3.926e-11, 5.464e-11,
00640
00641
                           7.012e-11, 8.912e-11, 1.127e-10, 1.347e-10, 1.498e-10, 1.544e-10,
                         1.602e-10, 1.824e-10, 2.078e-10, 2.366e-10, 2.691e-10, 5.141e-10, 8.259e-10, 1.254e-09, 1.849e-09, 2.473e-09, 3.294e-09, 4.16e-09, 5.095e-09, 6.11e-09, 6.93e-09, 7.888e-09, 8.903e-09, 9.713e-09, 1.052e-08, 1.115e-08, 1.173e-08, 1.21e-08, 1.228e-08, 1.239e-08,
00642
00643
00644
00646
                           1.231e-08, 1.213e-08, 1.192e-08, 1.138e-08, 1.085e-08, 1.008e-08,
                           9.224e-09, 8.389e-09, 7.262e-09, 6.278e-09, 5.335e-09, 4.388e-09,
00647
00648
                          3.589e-09, 2.761e-09, 2.129e-09, 1.633e-09, 1.243e-09, 9.681e-10,
                          8.355e-10, 7.665e-10, 7.442e-10, 8.584e-10, 9.732e-10, 1.063e-09,
00649
00650
                          1.163e-09, 1.286e-09, 1.472e-09, 1.707e-09, 2.032e-09, 2.474e-09,
00651
                           2.977e-09, 3.506e-09, 4.102e-09, 5.013e-09, 6.493e-09, 8.414e-09,
                           1.077e-08, 1.367e-08, 1.777e-08, 2.625e-08, 3.926e-08, 5.545e-08,
00653
                           7.195e-08, 9.464e-08, 1.404e-07, 2.183e-07, 3.329e-07, 4.535e-07,
                         6.158e-07, 8.187e-07, 1.075e-06, 1.422e-06, 1.979e-06, 2.71e-06, 3.58e-06, 4.573e-06, 5.951e-06, 7.999e-06, 1.072e-05, 1.372e-05, 1.697e-05, 2.112e-05, 2.643e-05, 3.288e-05, 3.994e-05, 4.794e-05, 5.606e-05, 6.383e-05, 7.286e-05, 8.156e-05, 8.883e-05, 9.469e-05,
00654
00655
00656
00657
                           9.848e-05, 0.0001023, 0.0001066, 0.0001115, 0.0001145, 0.0001142,
00659
                          0.0001133
00660
00661
                     static double no2[121] = {
00662
                          3.036e-09, 2.945e-10, 9.982e-11, 5.069e-11, 3.485e-11, 2.982e-11,
00663
                           2.947e-11, 3.164e-11, 3.714e-11, 4.586e-11, 6.164e-11, 8.041e-11, 9.982e-11, 1.283e-10, 1.73e-10, 2.56e-10, 3.909e-10, 5.959e-10,
00665
                           9.081e-10, 1.384e-09, 1.788e-09, 2.189e-09, 2.686e-09, 3.091e-09,
00666
                          3.49e-09, 3.796e-09, 4.2e-09, 5.103e-09, 6.005e-09, 6.3e-09, 6.706e-09, 7.07e-09, 7.434e-09, 7.663e-09, 7.788e-09, 7.8e-09, 7.597e-09, 7.482e-09, 7.227e-09, 6.403e-09, 5.585e-09, 4.606e-09, 3.703e-09, 2.984e-09, 2.183e-09, 1.48e-09, 8.441e-10, 5.994e-10, 3.799e-10,
00667
00668
00669
                           2.751e-10, 1.927e-10, 1.507e-10, 1.102e-10, 6.971e-11, 5.839e-11,
00671
00672
                           3.904e-11, 3.087e-11, 2.176e-11, 1.464e-11, 1.209e-11, 8.497e-12,
00673
                           6.477e-12, 4.371e-12, 2.914e-12, 2.424e-12, 1.753e-12, 1.35e-12,
00674
                          9.417e-13, 6.622e-13, 5.148e-13, 3.841e-13, 3.446e-13, 3.01e-13, 2.551e-13, 2.151e-13, 1.829e-13, 1.64e-13, 1.475e-13, 1.352e-13,
00675
                           1.155e-13, 9.963e-14, 9.771e-14, 9.577e-14, 9.384e-14, 9.186e-14,
00676
                            9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
                           9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
00678
00679
                           9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
00680
                          9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14
00681
00682
                     static double o3[121] = {
                           2.218e-08, 3.394e-08, 3.869e-08, 4.219e-08, 4.501e-08, 4.778e-08,
00684
00685
                           5.067e-08, 5.402e-08, 5.872e-08, 6.521e-08, 7.709e-08, 9.461e-08,
                          1.269e-07, 1.853e-07, 2.723e-07, 3.964e-07, 5.773e-07, 8.2e-07, 1.155e-06, 1.59e-06, 2.076e-06, 2.706e-06, 3.249e-06, 3.848e-06,
00686
00687
                           4.459e-06, 4.986e-06, 5.573e-06, 5.958e-06, 6.328e-06, 6.661e-06, 6.9e-06, 7.146e-06, 7.276e-06, 7.374e-06, 7.447e-06, 7.383e-06,
00688
                           7.321e-06, 7.161e-06, 6.879e-06, 6.611e-06, 6.216e-06, 5.765e-06,
00690
00691
                           5.355e-06, 4.905e-06, 4.471e-06, 4.075e-06, 3.728e-06, 3.413e-06,
00692
                           3.125e-06, 2.856e-06, 2.607e-06, 2.379e-06, 2.17e-06, 1.978e-06,
                           1.8e-06, 1.646e-06, 1.506e-06, 1.376e-06, 1.233e-06, 1.102e-06, 9.839e-07, 8.771e-07, 7.814e-07, 6.947e-07, 6.102e-07, 5.228e-07,
00693
00694
```

```
4.509e-07, 3.922e-07, 3.501e-07, 3.183e-07, 2.909e-07, 2.686e-07,
                        2.476e-07, 2.284e-07, 2.109e-07, 2.003e-07, 2.013e-07, 2.022e-07, 2.032e-07, 2.042e-07, 2.097e-07, 2.361e-07, 2.656e-07, 2.989e-07,
00696
00697
                         3.37e-07, 3.826e-07, 4.489e-07, 5.26e-07, 6.189e-07, 7.312e-07, 8.496e-07, 8.444e-07, 8.392e-07, 8.339e-07, 8.286e-07, 8.284e-07, 8.181e-07, 8.129e-07, 8.077e-07, 8.026e-07, 6.918e-07, 5.176e-07,
00698
00699
00700
00701
                         3.865e-07, 2.885e-07, 2.156e-07, 1.619e-07, 1.219e-07, 9.161e-08,
00702
                         6.972e-08, 5.399e-08, 3.498e-08, 2.111e-08, 1.322e-08, 8.482e-09,
00703
                         5.527e-09, 3.423e-09, 2.071e-09, 1.314e-09, 8.529e-10, 5.503e-10,
00704
                        3.665e-10
00705
00706
00707
                   static double ocs[121] = {
                         6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 5.997e-10,
00708
00709
                         5.989e-10, 5.881e-10, 5.765e-10, 5.433e-10, 5.074e-10, 4.567e-10,
00710
                         4.067e-10, 3.601e-10, 3.093e-10, 2.619e-10, 2.232e-10,
                                                                                                                                                                  1.805e-10,
                        1.46e-10, 1.187e-10, 8.03e-11, 5.435e-11, 3.686e-11, 2.217e-11, 1.341e-11, 8.756e-12, 4.511e-12, 2.37e-12, 1.264e-12, 8.28e-13,
00711
00712
                         5.263e-13, 3.209e-13, 1.717e-13, 9.068e-14, 4.709e-14, 2.389e-14,
00714
                         1.236e-14, 1.127e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00715
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00716
                        1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                        1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00717
00718
                        1.091e-14, 
00719
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00720
00721
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00722
                        1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00723
                        1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00724
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00725
                         1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00727
                        1.091e-14, 1.091e-14, 1.091e-14
00728
00729
00730
                   static double sf6[121] = {
                       4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12,
00731
                         4.103e-12, 4.103e-12, 4.103e-12, 4.087e-12, 4.064e-12, 4.023e-12,
00733
                         3.988e-12, 3.941e-12, 3.884e-12, 3.755e-12, 3.622e-12, 3.484e-12,
00734
                         3.32e-12, 3.144e-12, 2.978e-12, 2.811e-12, 2.653e-12, 2.489e-12,
00735
                        2.332e-12, 2.199e-12, 2.089e-12, 2.013e-12, 1.953e-12, 1.898e-12,
00736
                        1.859e-12, 1.826e-12, 1.798e-12, 1.776e-12, 1.757e-12, 1.742e-12,
                        1.728e-12, 1.717e-12, 1.707e-12, 1.698e-12, 1.691e-12, 1.685e-12, 1.679e-12, 1.675e-12, 1.671e-12, 1.668e-12, 1.665e-12, 1.663e-12,
00737
00738
00739
                         1.661e-12, 1.659e-12, 1.658e-12, 1.657e-12, 1.656e-12, 1.655e-12,
00740
                         1.654e-12, 1.653e-12, 1.653e-12, 1.652e-12, 1.652e-12, 1.652e-12,
00741
                        1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12,
00742
                        1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
                         1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-
00743
00744
00745
                         1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00746
                         1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00747
                         1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
                        1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12
00748
00749
00750
00751
00752
                   static double so2[121] = {
00753
                    1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10,
00754
                         1e-10, 1e-10, 9.867e-11, 9.537e-11, 9e-11, 8.404e-11, 7.799e-11,
                         7.205e-11, 6.616e-11, 6.036e-11, 5.475e-11, 5.007e-11, 4.638e-11,
00755
                         4.346e-11, 4.055e-11, 3.763e-11, 3.471e-11, 3.186e-11, 2.905e-11, 2.631e-11, 2.358e-11, 2.415e-11, 2.949e-11, 3.952e-11, 5.155e-11,
00756
                         6.76e-11, 8.741e-11, 1.099e-10, 1.278e-10, 1.414e-10, 1.512e-10,
00758
00759
                         1.607e-10, 1.699e-10, 1.774e-10, 1.832e-10, 1.871e-10, 1.907e-10
00760
                         1.943e-10, 1.974e-10, 1.993e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00761
                         2e-10, 2e
00762
00763
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00764
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00765
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00766
                         2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00767
                        2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10
00768
00769
00770
                   static int ig_co2 = -999;
00771
00772
                   double co2, *q[NG] = {NULL};
00773
00774
                   int ig. ip. iw. iz:
00775
00776
                    /\star Find emitter index of CO2... \star/
                   if (ig_co2 == -999)
  ig_co2 = find_emitter(ctl, "CO2");
00777
00778
00779
00780
                  /* Identify variable... */
00781
                   for (ig = 0; ig < ctl->ng; ig++) {
```

```
00782
          q[ig] = NULL;
00783
          if (strcasecmp(ctl->emitter[ig], "C2H2") == 0)
            q[ig] = c2h2;
00784
00785
          if (strcasecmp(ctl->emitter[ig], "C2H6") == 0)
00786
            q[ig] = c2h6;
00787
          if (strcasecmp(ctl->emitter[ig], "CCl4") == 0)
00788
            q[ig] = ccl4;
00789
          if (strcasecmp(ctl->emitter[ig], "CH4") == 0)
            q[ig] = ch4;
00790
          if (strcasecmp(ctl->emitter[ig], "ClO") == 0)
00791
00792
            q[ig] = clo;
00793
          if (strcasecmp(ctl->emitter[ig], "ClONO2") == 0)
            q[ig] = clono2;
00794
00795
          if (strcasecmp(ctl->emitter[ig], "CO") == 0)
00796
            q[ig] = co;
00797
          if (strcasecmp(ctl->emitter[ig], "COF2") == 0)
00798
            q[ig] = cof2;
00799
          if (strcasecmp(ctl->emitter[ig], "F11") == 0)
            q[ig] = f11;
00800
00801
             (strcasecmp(ctl->emitter[ig], "F12") == 0)
            q[ig] = f12;
00802
00803
          if (strcasecmp(ctl->emitter[ig], "F14") == 0)
          q[ig] = f14;
if (strcasecmp(ctl->emitter[ig], "F22") == 0)
00804
00805
00806
            q[ig] = f22;
           if (strcasecmp(ctl->emitter[ig], "H2O") == 0)
            q[ig] = h2o;
00808
00809
          if (strcasecmp(ctl->emitter[ig], "H2O2") == 0)
00810
            q[ig] = h2o2;
00811
          if (strcasecmp(ctl->emitter[iq], "HCN") == 0)
00812
            q[ig] = hcn;
00813
          if
             (strcasecmp(ctl->emitter[ig], "HNO3") == 0)
00814
            q[ig] = hno3;
00815
          if (strcasecmp(ctl->emitter[ig], "HNO4") == 0)
00816
            q[ig] = hno4;
          if (strcasecmp(ctl->emitter[ig], "HOC1") == 0)
00817
00818
            q[ig] = hocl;
          if (strcasecmp(ctl->emitter[ig], "N2O") == 0)
00820
            q[ig] = n2o;
00821
             (strcasecmp(ctl->emitter[ig], "N2O5") == 0)
00822
            q[ig] = n2o5;
          if (strcasecmp(ctl->emitter[ig], "NH3") == 0)
00823
00824
            q[ig] = nh3;
00825
          if (strcasecmp(ctl->emitter[ig], "NO") == 0)
00826
            q[ig] = no;
00827
          if
             (strcasecmp(ctl->emitter[ig], "NO2") == 0)
00828
            q[ig] = no2;
          if (strcasecmp(ctl->emitter[ig], "03") == 0)
00829
00830
            q[ig] = o3;
          if (strcasecmp(ctl->emitter[ig], "OCS") == 0)
00831
00832
            q[ig] = ocs;
00833
          if (strcasecmp(ctl->emitter[ig], "SF6") == 0)
00834
            q[ig] = sf6;
00835
          if (strcasecmp(ctl->emitter[ig], "SO2") == 0)
00836
            q[ig] = so2;
00837
00838
00839
        /* Loop over atmospheric data points... */
00840
        for (ip = 0; ip < atm->np; ip++) {
00841
00842
           /* Get altitude index... */
00843
          iz = locate(z, 121, atm \rightarrow z[ip]);
00844
00845
           /* Interpolate pressure... */
00846
          atm \rightarrow p[ip] = EXP(z[iz], pre[iz], z[iz + 1], pre[iz + 1], atm \rightarrow z[ip]);
00847
00848
          /* Interpolate temperature... */
00849
          atm \rightarrow t[ip] = LIN(z[iz], tem[iz], z[iz + 1], tem[iz + 1], atm \rightarrow z[ip]);
00850
00851
           /* Interpolate trace gases... */
00852
           for (ig = 0; ig < ctl->ng; ig++)
00853
             if (q[ig] != NULL)
00854
               atm->q[ig][ip]
                \label{eq:linear} \mbox{LIN(z[iz], q[ig][iz], z[iz + 1], q[ig][iz + 1], atm->z[ip]);}
00855
00856
            else
00857
              atm->q[ig][ip] = 0;
00858
           /* Set CO2... */
00859
00860
          if (ig_co2 >= 0) {
00861
            co2 =
              371.789948e-6 + 2.026214e-6 * (atm->time[ip] - 63158400.) / 31557600.;
00862
            atm \rightarrow q[ig\_co2][ip] = co2;
00863
00864
00865
00866
           /\star Set extinction to zero... \star/
          for (iw = 0; iw < ctl->nw; iw++)
  atm->k[iw][ip] = 0;
00867
00868
```

```
00869 }
00870 }
```

Here is the call graph for this function:



5.13.2.6 double ctmco2 ( double nu, double p, double t, double u)

Compute carbon dioxide continuum (optical depth).

Definition at line 874 of file jurassic.c.

```
00878
00879
            static double co2296[2001] = { 9.3388e-5, 9.7711e-5, 1.0224e-4, 1.0697e-4,
               1.1193e-4, 1.1712e-4, 1.2255e-4, 1.2824e-4, 1.3419e-4, 1.4043e-4,
00882
               1.4695e-4, 1.5378e-4, 1.6094e-4, 1.6842e-4, 1.7626e-4,
                                                                                                   1.8447e-4.
              1.33764, 1.33764, 1.33764, 1.33764, 1.33764, 1.33764, 1.33764, 2.3169-4, 2.02076-4, 2.11496-4, 2.21366-4, 2.31696-4, 2.42516-4, 2.53846-4, 2.6576-4, 2.78136-4, 2.91146-4, 3.04776-4, 3.19046-4, 3.33996-4, 3.49656-4, 3.66046-4, 3.83226-4, 4.01216-4, 4.20066-4, 4.3986-4, 4.60476-4, 4.82146-4, 5.04836-4, 5.2866-4, 5.5356-4,
00883
00884
00885
00887
               5.7959e-4, 6.0693e-4, 6.3557e-4, 6.6558e-4, 6.9702e-4, 7.2996e-4,
00888
               7.6449e-4, 8.0066e-4, 8.3856e-4, 8.7829e-4, 9.1991e-4, 9.6354e-4,
               .0010093, .0010572, .0011074, .00116, .0012152, .001273, .0013336, .0013972, .0014638, .0015336, .0016068, .0016835,
00889
00890
               .001764, .0018483, .0019367, .0020295, .0021267, .0022286, .0023355, .0024476, .0025652, .0026885, .0028178, .0029534, .0030956, .0032448, .0034012, .0035654, .0037375, .0039181,
00891
00892
00893
00894
               .0041076, .0043063, .0045148, .0047336, .0049632, .005204,
00895
               .0054567, .0057219, .0060002, .0062923, .0065988, .0069204,
               .007258, .0076123, .0079842, .0083746, .0087844, .0092146, .0096663, .01014, .010638, .011161, .01171, .012286, .012891, .013527, .014194, .014895, .015631, .016404, .017217, .01807, .018966, .019908, .020897, .021936, .023028, .024176, .025382, .026649, .027981, .02938, .030851, .032397, .034023, .035732,
00896
00897
00898
00899
00900
00901
               .037528, .039416, .041402, .04349, .045685, .047994, .050422,
               .052975, .055661, .058486, .061458, .064584, .067873, .071334, .074975, .078807, .082839, .087082, .091549, .096249, .1012, .10641, .11189, .11767, .12375, .13015, .13689, .14399, .15147, .15935, .16765, .17639, .18561, .19531, .20554, .21632, .22769,
00902
00903
00904
00905
00906
               .23967, .25229, .2656, .27964, .29443, .31004, .3265, .34386,
00907
               .36218, .3815, .40188, .42339, .44609, .47004, .49533, .52202
               .5502, .57995, .61137, .64455, .6796, .71663, .75574, .79707, .84075, .88691, .9357, .98728, 1.0418, 1.0995, 1.1605, 1.225, 1.2932, 1.3654, 1.4418, 1.5227, 1.6083, 1.6989, 1.7948, 1.8964,
00908
00909
00910
               2.004, 2.118, 2.2388, 2.3668, 2.5025, 2.6463, 2.7988, 2.9606,
00911
               3.1321, 3.314, 3.5071, 3.712, 3.9296, 4.1605, 4.4058,
00912
                                                                                                 4.6663,
00913
               4.9431, 5.2374, 5.5501, 5.8818, 6.2353, 6.6114, 7.0115, 7.4372
00914
               7.8905, 8.3731, 8.8871, 9.4349, 10.019, 10.641, 11.305, 12.013,
               12.769, 13.576, 14.437, 15.358, 16.342, 17.39, 18.513, 19.716,
00915
00916
               21.003, 22.379, 23.854, 25.436, 27.126, 28.942, 30.89, 32.973,
               35.219, 37.634, 40.224, 43.021, 46.037, 49.29, 52.803, 56.447,
00918
               60.418, 64.792, 69.526, 74.637, 80.182, 86.193, 92.713, 99.786,
00919
               107.47, 115.84, 124.94, 134.86, 145.69, 157.49, 170.3, 184.39,
               199.83, 216.4, 234.55, 254.72, 276.82, 299.85, 326.16, 354.99,
00920
               386.51, 416.68, 449.89, 490.12, 534.35, 578.25, 632.26, 692.61, 756.43, 834.75, 924.11, 1016.9, 996.96, 1102.7, 1219.2, 1351.9,
00921
00922
00923
               1494.3, 1654.1, 1826.5, 2027.9, 2249., 2453.8, 2714.4, 2999.4,
               3209.5, 3509., 3840.4, 3907.5, 4190.7, 4533.5, 4648.3, 5059.1,
```

```
5561.6, 6191.4, 6820.8, 7905.9, 9362.2, 2431.3, 2211.3, 2046.8,
                 2023.8, 1985.9, 1905.9, 1491.1, 1369.8, 1262.2, 1200.7, 887.74, 820.25, 885.23, 887.21, 816.73, 1126.9, 1216.2, 1272.4, 1579.5,
00926
00927
                 1634.2, 1656.3, 1657.9, 1789.5, 1670.8, 1509.5, 8474.6, 7489.2, 6793.6, 6117., 5574.1, 5141.2, 5084.6, 4745.1, 4413.2, 4102.8, 4024.7, 3715., 3398.6, 3100.8, 2900.4, 2629.2, 2374., 2144.7,
00928
00929
00930
                 1955. 8, 1760.8, 1591.2, 1435.2, 1296.2, 1174., 1065.1, 967.76, 999.48, 897.45, 809.23, 732.77, 670.26, 611.93, 560.11, 518.77
00932
00933
                 476.84, 438.8, 408.48, 380.21, 349.24, 322.71, 296.65, 272.85,
                 251.96, 232.04, 213.88, 197.69, 182.41, 168.41, 155.79, 144.05, 133.31, 123.48, 114.5, 106.21, 98.591, 91.612, 85.156, 79.204, 73.719, 68.666, 63.975, 59.637, 56.35, 52.545, 49.042, 45.788, 42.78, 39.992, 37.441, 35.037, 32.8, 30.744, 28.801, 26.986, 25.297, 23.731, 22.258, 20.883, 19.603, 18.403, 17.295, 16.249,
00934
00935
00936
00937
00938
00939
                 15.271, 14.356, 13.501, 12.701, 11.954, 11.254, 10.6, 9.9864,
                 9.4118, 8.8745, 8.3714, 7.8997, 7.4578, 7.0446, 6.6573, 6.2949, 5.9577, 5.6395, 5.3419, 5.063, 4.8037, 4.5608, 4.3452, 4.1364, 3.9413, 3.7394, 3.562, 3.3932, 3.2325, 3.0789, 2.9318, 2.7898, 2.6537, 2.5225, 2.3958, 2.2305, 2.1215, 2.0245, 1.9427, 1.8795,
00940
00941
00942
                 1.8336, 1.7604, 1.7016, 1.6419, 1.5282, 1.4611, 1.3443, 1.27,
00944
                 1.1675, 1.0824, 1.0534, .99833, .95854, .92981, .90887, .89346,
00945
00946
                 .88113, .87068, .86102, .85096, .88262, .86151, .83565, .80518,
00947
                  .77045, .73736, .74744, .74954, .75773, .82267, .83493, .89402,
                 .89725, .93426, .95564, .94045, .94174, .93404, .92035, .90456, .88621, .86673, .78117, .7515, .72056, .68822, .65658, .62764, .55984, .55598, .57407, .60963, .63763, .66198, .61132, .60972,
00948
00949
                 .52496, .50649, .41872, .3964, .32422, .27276, .24048, .23772,
00951
00952
                  .2286, .22711, .23999, .32038, .34371, .36621, .38561, .39953,
                 .40636, .44913, .42716, .3919, .35477, .33935, .3351, .39746, .40993, .49398, .49956, .56157, .54742, .57295, .57386, .55417,
00953
00954
                 .50745, .471, .43446, .39102, .34993, .31269, .27888, .24912,
00955
                 .22291, .19994, .17972, .16197, .14633, .13252, .12029, .10942,
                 .099745, .091118, .083404, .076494, .070292, .064716, .059697, .055173, .051093, .047411, .044089, .041092, .038392, .035965,
00957
00958
00959
                 .033789, .031846, .030122, .028607, .02729, .026169, .025209,
00960
                 .024405, .023766, .023288, .022925, .022716, .022681, .022685,
                 .022768, .023133, .023325, .023486, .024004, .024126, .024083, .023785, .024023, .023029, .021649, .021108, .019454, .017809, .017292, .016635, .017037, .018068, .018977, .018756, .017847,
00961
00963
                 .016557, .016142, .014459, .012869, .012381, .010875, .0098701
00964
                 .009285, .0091698, .0091701, .0096145, .010553, .01106, .012613, .014362, .015017, .016507, .017741, .01768, .017784, .0171,
00965
00966
                .016357, .016172, .017257, .018978, .020935, .021741, .023567, .025183, .025589, .026732, .027648, .028278, .028215, .02856, .029015, .029062, .028851, .028497, .027825, .027801, .026523, .02487, .022967, .022168, .020194, .018605, .017903, .018439,
00967
00968
00970
                 .019697, .020311, .020855, .020057, .018608, .016738, .015963, .013844, .011801, .011134, .0097573, .0086007, .0086226, .0083721, .0090978, .0097616, .0098426, .011317, .012853, .014 .014657, .015771, .016351, .016079, .014829, .013431, .013185, .013207, .01448, .016176, .017971, .018265, .019526, .020455,
00971
00972
00973
                                                                                                                        .01447.
00974
                 .019797, .019802, .0194, .018176, .017505, .016197, .015339, .014401, .013213, .012203, .011186, .010236, .0093288, .0084854,
00976
00977
                 .0076837, .0069375, .0062614, .0056628, .0051153, .0046015, .0041501, .003752, .0033996, .0030865, .0028077, .0025586,
00978
00979
                 .0023355, .0021353, .0019553, .0017931, .0016466, .0015141, .0013941, .0012852, .0011862, .0010962, .0010142, 9.3935e-4, 8.71e-4, 8.0851e-4, 7.5132e-4, 6.9894e-4, 6.5093e-4, 6.0689e-4,
00980
00982
                 5.6647e-4, 5.2935e-4, 4.9525e-4, 4.6391e-4, 4.3509e-4, 4.086e-4,
00983
00984
                 3.8424e-4, 3.6185e-4, 3.4126e-4, 3.2235e-4, 3.0498e-4, 2.8904e-4,
00985
                 2.7444e-4, 2.6106e-4, 2.4883e-4, 2.3766e-4, 2.275e-4, 2.1827e-4,
                 2.0992-4, 2.0239-4, 1.9563e-4, 1.896e-4, 1.8427e-4, 1.796e-4, 1.7555e-4, 1.7209e-4, 1.692e-4, 1.6687e-4, 1.6505e-4, 1.6375e-4,
00986
                 1.6294e-4, 1.6261e-4, 1.6274e-4, 1.6334e-4, 1.6438e-4, 1.6587e-4,
00988
00989
                 1.678e-4, 1.7017e-4, 1.7297e-4, 1.762e-4, 1.7988e-4, 1.8399e-4,
00990
                 1.8855e-4, 1.9355e-4, 1.9902e-4, 2.0494e-4, 2.1134e-4, 2.1823e-4,
                 2.2561e-4, 2.335e-4, 2.4192e-4, 2.5088e-4, 2.604e-4, 2.705e-4, 2.8119e-4, 2.9251e-4, 3.0447e-4, 3.171e-4, 3.3042e-4, 3.4447e-4, 3.5927e-4, 3.7486e-4, 3.9127e-4, 4.0854e-4, 4.267e-4, 4.4579e-4,
00991
00992
                  4.6586e-4, 4.8696e-4, 5.0912e-4, 5.324e-4, 5.5685e-4, 5.8253e-4,
                 6.0949e-4, 6.378e-4, 6.6753e-4, 6.9873e-4, 7.3149e-4, 7.6588e-4,
00995
00996
                 8.0198e-4, 8.3987e-4, 8.7964e-4, 9.2139e-4, 9.6522e-4, .0010112,
                  .0010595, .0011102, .0011634, .0012193, .001278, .0013396,
00997
                 .0014043, .0014722, .0015436, .0016185, .0016972, .0017799, .0018668, .001958, .0020539, .0021547, .0022606, .0023719, .002489, .002612, .0027414, .0028775, .0030206, .0031712,
00998
00999
                 .0033295, .0034962, .0036716, .0038563, .0040506, .0042553,
01001
                 .0044709, .004698, .0049373, .0051894, .0054552, .0057354, .006031, .0063427, .0066717, .0070188, .0073854, .0077726,
01002
01003
                 .0081816, .0086138, .0090709, .0095543, .010066, .010607, .011181, .011789, .012433, .013116, .013842, .014613, .015432, .016304, .017233, .018224, .019281, .020394, .021574, .022836,
01004
01005
                 .024181, .025594, .027088, .028707, .030401, .032245, .034219, .036262, .038539, .040987, .043578, .04641, .04949, .052726,
01007
01008
                 .056326, .0602, .064093, .068521, .073278, .077734, .083064, .088731, .093885, .1003, .1072, .11365, .12187, .13078, .13989, .15095, .16299, .17634, .19116, .20628, .22419, .24386, .26587,
01009
01010
01011
```

```
.28811, .31399, .34321, .36606, .39675, .42742, .44243, .47197,
                .49993, .49027, .51147, .52803, .48931, .49729, .5026, .43854,
01013
01014
                 .441, .44766, .43414, .46151, .50029, .55247, .43855, .32115,
01015
                 .32607, .3431, .36119, .38029, .41179, .43996, .47144, .51853,
                .55362, .59122, .66338, .69877, .74001, .82923, .86907, .90361, 1.0025, 1.031, 1.0559, 1.104, 1.1178, 1.1341, 1.1547, 1.351,
01016
                1.4772, 1.4812, 1.4907, 1.512, 1.5442, 1.5853, 1.6358, 1.6963,
                1.7674, 1.8474, 1.9353, 2.0335, 2.143, 2.2592, 2.3853, 2.5217,
01019
01020
                2.6686, 2.8273, 2.9998, 3.183, 3.3868, 3.6109, 3.8564, 4.1159,
                2.686, 2.82/3, 2.9998, 3.183, 3.3868, 3.6109, 3.8564, 4.1159, 4.4079, 4.7278, 5.0497, 5.3695, 5.758, 6.0834, 6.4976, 6.9312, 7.38, 7.5746, 7.9833, 8.3791, 8.3956, 8.7501, 9.1067, 9.072, 9.4649, 9.9112, 10.402, 10.829, 11.605, 12.54, 12.713, 10.443, 10.825, 11.375, 11.955, 12.623, 13.326, 14.101, 15.041, 15.547, 16.461, 17.439, 18.716, 19.84, 21.036, 22.642, 23.901, 25.244, 27.03, 28.411, 29.871, 31.403, 33.147, 34.744, 36.456, 39.239, 45.162, 47.004, 49.032, 51.301, 53.046, 56.73, 59.632
01021
01022
01023
01025
01026
                43.605, 45.162, 47.004, 49.093, 51.391, 53.946, 56.673, 59.629, 63.167, 66.576, 70.254, 74.222, 78.477, 83.034, 87.914, 93.18, 98.77, 104.74, 111.15, 117.95, 125.23, 133.01, 141.33, 150.21, 159.71, 169.89, 180.93, 192.54, 204.99, 218.34, 232.65, 248.,
01027
01028
01029
                264.47, 282.14, 301.13, 321.53, 343.48, 367.08, 392.5, 419.88,
                 449.4, 481.26, 515.64, 552.79, 592.99, 636.48, 683.61, 734.65,
01032
01033
                789.99, 850.02, 915.14, 985.81, 1062.5, 1147.1, 1237.8, 1336.4,
                1443.2, 1558.9, 1684.2, 1819.2, 1965.2, 2122.6, 2291.7, 2470.8, 2665.7, 2874.9, 3099.4, 3337.9, 3541., 3813.3, 4111.9, 4439.3, 4798.9, 5196., 5639.2, 6087.5, 6657.7, 7306.7, 8040.7, 8845.5,
01034
01035
01036
                9702.2, 10670., 11739., 12842., 14141., 15498., 17068., 18729., 20557., 22559., 25248., 27664., 30207., 32915., 35611., 38081.,
01038
01039
                40715., 43191., 41651., 42750., 43785., 44353., 44366., 44189.,
01040
                43618., 42862., 41878., 35133., 35215., 36383., 39420., 44055.,
                44155., 45850., 46853., 39197., 38274., 29942., 28553., 21792.,
01041
01042
                21228., 17106., 14955., 18181., 19557., 21427., 23728., 26301.,
                28584., 30775., 32536., 33867., 40089., 39204., 37329., 34452.,
                31373., 33921., 34800., 36043., 44415., 45162., 52181., 50895.,
01044
01045
                54140., 50840., 50468., 48302., 44915., 40910., 36754., 32755.,
                29093., 25860., 22962., 20448., 18247., 16326., 14645., 13165., 11861., 10708., 9686.9, 8779.7, 7971.9, 7250.8, 6605.7, 6027.2, 5507.3, 5039.1, 4616.6, 4234.8, 3889., 3575.4, 3290.5, 3031.3, 2795.2, 2579.9, 2383.1, 2203.3, 2038.6, 1887.6, 1749.1, 1621.9,
01046
01047
01048
01050
                1505., 1397.4, 1298.3, 1207., 1122.8, 1045., 973.1, 906.64,
                845.16, 788.22, 735.48, 686.57, 641.21, 599.1, 559.99, 523.64, 489.85, 458.42, 429.16, 401.92, 376.54, 352.88, 330.82, 310.24,
01051
01052
                291.03, 273.09, 256.34, 240.69, 226.05, 212.37, 199.57, 187.59,
01053
                176.37, 165.87, 156.03, 146.82, 138.17, 130.07, 122.47, 115.34, 108.65, 102.37, 96.473, 90.934, 85.73, 80.84, 76.243, 71.922,
01054
                67.858, 64.034, 60.438, 57.052, 53.866, 50.866, 48.04, 45.379, 42.872, 40.51, 38.285, 36.188, 34.211, 32.347, 30.588, 28.929,
01057
01058
                27.362, 25.884, 24.489, 23.171, 21.929, 20.755, 19.646, 18.599,
                17.61, 16.677, 15.795, 14.961, 14.174, 13.43, 12.725, 12.06, 11.431, 10.834, 10.27, 9.7361, 9.2302, 8.7518, 8.2997, 7.8724,
01059
01060
                7.4674, 7.0848, 6.7226, 6.3794, 6.054, 5.745, 5.4525, 5.1752,
01061
                 4.9121, 4.6625, 4.4259, 4.2015, 3.9888, 3.7872, 3.5961, 3.4149,
                3.2431, 3.0802, 2.9257, 2.7792, 2.6402, 2.5084, 2.3834, 2.2648,
01063
01064
                2.1522, 2.0455, 1.9441, 1.848, 1.7567, 1.6701, 1.5878, 1.5097,
                1.4356, 1.3651, 1.2981, 1.2345, 1.174, 1.1167, 1.062, 1.0101, .96087, .91414, .86986, .82781, .78777, .74971, .71339, .67882, .64604, .61473, .58507, .55676, .52987, .5044, .48014, .45715, .43527, .41453, .3948, .37609, .35831, .34142, .32524, .30995, .29536, .28142, .26807, .25527, .24311, .23166, .22077, .21053,
01065
01066
01067
01069
                .20081, .19143, .18261, .17407, .16603, .15833, .15089, .14385
01070
                13707, 13065, .12449, .11865, .11306, .10774, .10266, .097818, .093203, .088815, .084641, .080671, .076892, .073296, .069873, .066613, .06351, .060555, .05774, .055058, .052504, .050071, .047752, .045543, .043438, .041432, .039521, .037699, .035962,
01071
01072
01073
                .034307, .032729, .031225, .029791, .028423, .02712, .025877, .024692, .023563, .022485, .021458, .020478, .019543, .018652,
01075
01076
01077
                 .017802, .016992, .016219, .015481, .014778, .014107, .013467,
                .012856, .012274, .011718, .011188, .010682, .0102, .0097393, .0093001, .008881, .0084812, .0080997, .0077358, .0073885, .0070571, .0067409, .0064393, .0061514, .0058768, .0056147, .0053647, .0051262, .0048987, .0046816, .0044745, .0042769,
01078
01079
01080
                .0040884, .0039088, .0037373, .0035739, .003418, .0032693,
01082
01083
                .0031277, .0029926, .0028639, .0027413, .0026245, .0025133,
                 .0024074, .0023066, .0022108, .0021196, .002033, .0019507,
01084
                .0018726, .0017985, .0017282, .0016617, .0015988, .0015394,
01085
                .0014834, .0014306, .0013811, .0013346, .0012911, .0012506, .0012131, .0011784, .0011465, .0011175, .0010912, .0010678, .0010472, .0010295, .0010147, .001003, 9.9428e-4, 9.8883e-4,
01086
01088
                9.8673e-4, 9.8821e-4, 9.9343e-4, .0010027, .0010164, .0010348, .0010586, .0010882, .0011245, .0011685, .0012145, .0012666,
01089
01090
                 .0013095, .0013688, .0014048, .0014663, .0015309, .0015499,
01091
                .0016144, .0016312, .001705, .0017892, .0018499, .0019715, .0021102, .0022442, .0024284, .0025893, .0027703, .0029445,
01092
                .0031193, .003346, .0034552, .0036906, .0037584, .0040084, .0041934, .0044587, .0047093, .0049759, .0053421, .0055134,
01094
01095
01096
                 .0059048, \ .0058663, \ .0061036, \ .0063259, \ .0059657, \ .0060653,
                .0060972, .0055539, .0055653, .0055772, .005331, .0054953, .0055919, .0058684, .006183, .0066675, .0069808, .0075142,
01097
01098
```

```
.0078536, .0084282, .0089454, .0094625, .0093703, .0095857,
                .0099283, .010063, .010521, .0097778, .0098175, .010379, .010447,
01100
                 .0105, .010617, .010706, .01078, .011177, .011212, .011304,
01101
                 .011446, .011603, .011816, .012165, .012545, .013069, .013539,
01102
               .01411, .014776, .016103, .017016, .017994, .018978, .01998, .021799, .022745, .023681, .024627, .025562, .026992, .027958, .029013, .030154, .031402, .03228, .033651, .035272, .037088, .039021, .041213, .043597, .045977, .04877, .051809, .054943,
01103
01104
01106
               .039021, .041213, .043597, .045977, .04877, .051809, .054943, .058064, .061528, .06537, .069309, .071928, .075752, .079589, .083352, .084096, .087497, .090817, .091198, .094966, .099045, .10429, .10867, .11518, .12269, .13126, .14087, .15161, .16388, .16423, .1759, .18721, .19994, .21275, .22513, .23041, .24231, .25299, .25396, .26396, .27696, .27929, .2908, .30595, .31433, .3282, .3429, .35944, .37467, .39277, .41245, .43326, .45649,
01107
01108
01109
01110
01111
01112
01113
                .48152, .51897, .54686, .57877, .61263, .64962, .68983, .73945
                .78619, .83537, .89622, .95002, 1.0067, 1.0742, 1.1355, 1.2007, 1.2738, 1.347, 1.4254, 1.5094, 1.6009, 1.6976, 1.8019, 1.9148, 2.0357, 2.166, 2.3066, 2.4579, 2.6208, 2.7966, 2.986, 3.188,
01114
01115
01116
                3.4081, 3.6456, 3.9, 4.1747, 4.4712, 4.7931, 5.1359, 5.5097,
                5.9117, 6.3435, 6.8003, 7.3001, 7.8385, 8.3945, 9.011, 9.6869,
                10.392, 11.18, 12.036, 12.938, 13.944, 14.881, 16.029, 17.2
18.574, 19.945, 21.38, 22.9, 24.477, 26.128, 27.87, 29.037,
01120
                30.988, 33.145, 35.506, 37.76, 40.885, 44.487, 48.505, 52.911, 57.56, 61.964, 67.217, 72.26, 78.343, 85.08, 91.867, 99.435, 107.68, 116.97, 127.12, 138.32, 150.26, 163.04, 174.81, 189.26,
01121
01122
01123
                205.61, 224.68, 240.98, 261.88, 285.1, 307.58, 334.35, 363.53,
01124
                394.68, 427.85, 458.85, 489.25, 472.87, 486.93, 496.27, 501.52
01125
01126
                501.57, 497.14, 488.09, 476.32, 393.76, 388.51, 393.42, 414.45,
                455.12, 514.62, 520.38, 547.42, 562.6, 487.47, 480.83, 391.06, 376.92, 303.7, 295.91, 256.03, 236.73, 280.38, 310.71, 335.53, 367.88, 401.94, 435.52, 469.13, 497.94, 588.82, 597.94, 597.2, 588.28, 571.2, 555.75, 603.56, 638.15, 680.75, 801.72, 848.01,
01127
01128
01129
                962.15, 990.06, 1068.1, 1076.2, 1115.3, 1134.2, 1136.6, 1119.1
01131
01132
                1108.9, 1090.6, 1068.7, 1041.9, 1005.4, 967.98, 927.08, 780.1,
01133
                751.41, 733.12, 742.65, 785.56, 855.16, 852.45, 878.1, 784.59,
                777.81, 765.13, 622.93, 498.09, 474.89, 386.9, 378.48, 336.17, 322.04, 329.57, 350.5, 383.38, 420.02, 462.39, 499.71, 531.98, 654.99, 653.43, 639.99, 605.16, 554.16, 504.42, 540.64, 552.33,
01134
01135
                 679.46, 699.51, 713.91, 832.17, 919.91, 884.96, 907.57, 846.56,
                818.56, 768.93, 706.71, 642.17, 575.95, 515.38, 459.07, 409.02, 364.61, 325.46, 291.1, 260.89, 234.39, 211.01, 190.38, 172.11,
01138
01139
                155.91, 141.49, 128.63, 117.13, 106.84, 97.584, 89.262, 81.756, 74.975, 68.842, 63.28, 58.232, 53.641, 49.46, 45.649, 42.168, 38.991, 36.078, 33.409, 30.96, 28.71, 26.642, 24.737, 22.985,
01140
01141
01142
                21.37, 19.882, 18.512, 17.242, 16.073, 14.987, 13.984, 13.05,
01144
                12.186, 11.384, 10.637, 9.9436, 9.2988, 8.6991, 8.141, 7.6215
                7.1378, 6.6872, 6.2671, 5.8754, 5.51, 5.1691, 4.851, 4.5539, 4.2764, 4.0169, 3.7742, 3.5472, 3.3348, 3.1359, 2.9495, 2.7749,
01145
01146
                2.6113, 2.4578, 2.3139, 2.1789, 2.0523, 1.9334, 1.8219, 1.7171,
01147
                1.6188, 1.5263, 1.4395, 1.3579, 1.2812, 1.209, 1.1411, 1.0773,
01148
                1.0171, .96048, .90713, .85684, .80959, .76495, .72282, .68309,
                .64563, .61035, .57707, .54573, .51622, .48834, .46199, .43709,
01150
01151
                 .41359, .39129, .37034, .35064, .33198, .31442, .29784,
01152
                 .26732, .25337, .24017, .22774, .21601, .20479, .19426
01153
01154
            static double co2260[2001] = { 5.7971e-5, 6.0733e-5, 6.3628e-5, 6.6662e-5,
                6.9843e-5, 7.3176e-5, 7.6671e-5, 8.0334e-5, 8.4175e-5, 8.8201e-5,
01156
                9.2421e-5, 9.6846e-5, 1.0149e-4, 1.0635e-4, 1.1145e-4,
01157
01158
                1.224e-4, 1.2828e-4, 1.3444e-4, 1.409e-4, 1.4768e-4, 1.5479e-4,
01159
                1.6224e-4, 1.7006e-4, 1.7826e-4, 1.8685e-4, 1.9587e-4, 2.0532e-4,
                2.1524e-4, 2.2565e-4, 2.3656e-4, 2.48e-4, 2.6001e-4, 2.7261e-4, 2.8582e-4, 2.9968e-4, 3.1422e-4, 3.2948e-4, 3.4548e-4, 3.6228e-4,
01160
                3.799e-4, 3.9838e-4, 4.1778e-4, 4.3814e-4, 4.595e-4, 4.8191e-4,
                5.0543e-4, 5.3012e-4, 5.5603e-4, 5.8321e-4, 6.1175e-4, 6.417e-4,
01163
01164
                6.7314e-4, 7.0614e-4, 7.4078e-4, 7.7714e-4, 8.1531e-4, 8.5538e-4,
01165
                8.9745e-4, 9.4162e-4, 9.8798e-4, .0010367, .0010878, .0011415,
                .0011978, .001257, .0013191, .0013844, .001453, .0015249, .0016006, .00168, .0017634, .001851, .001943, .0020397, .0021412, .0022479, .00236, .0024778, .0026015, .0027316, .0028682,
01166
01167
                .0030117, .0031626, .0033211, .0034877, .0036628, .0038469, .0040403, .0042436, .0044574, .004682, .0049182, .0051665,
01169
01170
                .0054276, .0057021, .0059907, .0062942, .0066133, .0069489, .0073018, .0076729, .0080632, .0084738, .0089056, .0093599,
01171
01172
                .0098377, .01034, .010869, .011426, .012011, .012627, .013276, .013958, .014676, .015431, .016226, .017063, .017944, .018872, .019848, .020876, .021958, .023098, .024298, .025561, .026892,
01173
01175
                .028293, .029769, .031323, .032961, .034686, .036503, .038418, .040435, .042561, .044801, .047161, .049649, .052271, .055035,
01176
01177
                .047161, .043643, .042361, .0447161, .043643, .032211, .033033, .057948, .061019, .064256, .06767, .07127, .075066, .079069, .083291, .087744, .092441, .097396, .10262, .10814, .11396, .1201, .12658, .13342, .14064, .14826, .1563, .1648, .17376, .18323, .19324, .2038, .21496, .22674, .23919, .25234, .26624, .28093, .29646, .31287, .33021, .34855, .36794, .38844, .41012, .2674
01178
01179
01182
01183
                 .43305, .45731, .48297, .51011, .53884, .56924, .60141, .63547,
                .67152, .70969, .75012, .79292, .83826, .8863, .93718, .99111, 1.0482, 1.1088, 1.173, 1.2411, 1.3133, 1.3898, 1.471, 1.5571,
01184
01185
```

```
1.6485, 1.7455, 1.8485, 1.9577, 2.0737, 2.197, 2.3278, 2.4668,
                   2.6145, 2.7715, 2.9383, 3.1156, 3.3042, 3.5047, 3.7181, 3.9451, 4.1866, 4.4437, 4.7174, 5.0089, 5.3192, 5.65, 6.0025, 6.3782,
01188
                    6.7787, 7.206, 7.6617, 8.1479, 8.6669, 9.221, 9.8128, 10.445,
01189
                   11.12, 11.843, 12.615, 13.441, 14.325, 15.271, 16.283, 17.367 18.529, 19.776, 21.111, 22.544, 24.082, 25.731, 27.504, 29.40
01190
01191
                    31.452, 33.654, 36.024, 38.573, 41.323, 44.29, 47.492, 50.951,
01192
                    54.608, 58.588, 62.929, 67.629, 72.712, 78.226, 84.207, 90.699
01193
                   97.749, 105.42, 113.77, 122.86, 132.78, 143.61, 155.44, 168.33, 182.48, 198.01, 214.87, 233.39, 253.86, 276.34, 300.3, 327.28,
01194
01195
                    356.89, 389.48, 422.29, 458.99, 501.39, 548.13, 595.62, 652.74,
01196
                   716.54, 784.57, 866.78, 960.59, 1062.8, 1072.5, 1189.5, 1319.4, 1467.6, 1630.2, 1813.7, 2016.9, 2253., 2515.3, 2773.5, 3092.8, 3444.4, 3720.4, 4104.3, 4527.5, 4645.9, 5021.7, 5462.2, 5597.,
01197
01198
01199
01200
                    6110.6, 6732.5, 7513.8, 8270.6, 9640.6, 11487., 2796.1, 2680.1,
                   2441.6, 2404.2, 2334.8, 2215.2, 1642.5, 1477.9, 1328.1, 1223.5, 843.34, 766.96, 831.65, 834.84, 774.85, 1156.3, 1275.6, 1366.1, 1795.6, 1885., 1936.5, 1953.4, 2154.4, 2002.7, 1789.8, 10381., 9040., 8216.5, 7384.7, 6721.9, 6187.7, 6143.8, 5703.9, 5276.6,
01201
01202
01203
                    4873.1, 4736., 4325.3, 3927., 3554.1, 3286.1, 2950.1, 2642.4,
01205
                    2368.7, 2138.9, 1914., 1719.6, 1543.9, 1388.6, 1252.1, 1132.2
01206
01207
                    1024.1, 1025.4, 920.58, 829.59, 750.54, 685.01, 624.25, 570.14,
                   525.81, 481.85, 441.95, 408.71, 377.23, 345.86, 318.51, 292.26,
01208
                   268.34, 247.04, 227.14, 209.02, 192.69, 177.59, 163.78, 151.26, 139.73, 129.19, 119.53, 110.7, 102.57, 95.109, 88.264, 81.948, 76.13, 70.768, 65.827, 61.251, 57.022, 53.495, 49.824, 46.443,
01209
01210
01211
                    43.307, 40.405, 37.716, 35.241, 32.923, 30.77, 28.78, 26.915,
01212
01213
                   25.177, 23.56, 22.059, 20.654, 19.345, 18.126, 16.988, 15.93,
01214
                   14.939, 14.014, 13.149, 12.343, 11.589, 10.884, 10.225, 9.6093,
01215
                    9.0327, 8.4934, 7.9889, 7.5166, 7.0744, 6.6604, 6.2727, 5.9098,
                   5.5701, 5.2529, 4.955, 4.676, 4.4148, 4.171, 3.9426, 3.7332,
01216
                    3.5347, 3.3493, 3.1677, 3.0025, 2.8466, 2.6994, 2.5601, 2.4277
                    2.3016, 2.1814, 2.0664, 1.9564, 1.8279, 1.7311, 1.6427, 1.5645,
01218
01219
                   1.4982, 1.443, 1.374, 1.3146, 1.2562, 1.17, 1.1105, 1.0272,
                  . 96863, .89718, .83654, .80226, .75908, .72431, .69573, .67174, .65126, .63315, .61693, .60182, .58715, .59554, .57649, .55526, .53177, .50622, .48176, .4813, .47642, .47492, .50273, .50293, .52687, .52239, .53419, .53814, .52626, .52211, .51492, .50622, .49746, .48841, .4792, .43534, .41999, .40349, .38586, .36799, .55626, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627, .56627,
01220
01221
01222
01224
                   .35108, .31089, .30803, .3171, .33599, .35041, .36149, .32924, .32462, .27309, .25961, .20922, .19504, .15683, .13098, .11588,
01225
01226
                    .11478, .11204, .11363, .12135, .16423, .17785, .19094, .20236,
01227
                  .21084, .2154, .24108, .22848, .20871, .18797, .17963, .17834, .21552, .22284, .26945, .27052, .30108, .28977, .29772, .29224, .27658, .24956, .22777, .20654, .18392, .16338, .1452, .12916, .1152, .10304, .092437, .083163, .075031, .067878, .061564,
01228
01230
01231
01232
                    .055976, .051018, .046609, .042679, .03917, .036032, .033223,
                   .030706, .02845, .026428, .024617, .022998, .021554, .02027, .019136, .018141, .017278, .016541, .015926, .015432, .015058, .014807, .014666, .014635, .014728, .014947, .01527, .015728, .016345, .017026, .017798, .018839, .019752, .020636, .021886,
01233
01234
01235
                   .022695, .02327, .023478, .024292, .023544, .022222, .021932, .020052, .018143, .017722, .017031, .017782, .01938, .020734,
01237
01238
                   .020476, .019255, .017477, .016878, .014617, .012489, .011765, .0099077, .0086446, .0079446, .0078644, .0079763, .008671,
01239
01240
                   .01001, .0108, .012933, .015349, .016341, .018484, .020254, .020254, .020254, .020478, .019591, .018595, .018385, .019913, .022254, .024847, .025809, .028053, .029924, .030212, .031367, .03222,
01241
01243
                   .032739, .032537, .03286, .033344, .033507, .033499, .033339, .032809, .033041, .031723, .029837, .027511, .026603, .024032,
01244
01245
01246
                    .021914, .020948, .021701, .023425, .024259, .024987, .023818,
01247
                   .021768, .019223, .018144, .015282, .012604, .01163, .0097907,
                   .008336, .0082473, .0079582, .0088077, .009779, .010129, .012145,
                   .014378, .016761, .01726, .018997, .019998, .019809, .01819, .016358, .016099, .01617, .017939, .020223, .022521, .02277,
01249
01250
01251
                    .024279, .025247, .024222, .023989, .023224, .021493, .020362,
                   .018596, .017309, .015975, .014466, .013171, .011921, .01078, .0097229, .0087612, .0078729, .0070682, .0063494, .0057156, .0051459, .0046273, .0041712, .0037686, .0034119, .003095,
01252
01253
                    .0028126, .0025603, .0023342, .0021314, .0019489,
                    .001636, .0015017, .00138, .0012697, .0011694, .0010782,
01256
01257
                    9.9507e-4, 9.1931e-4, 8.5013e-4, 7.869e-4, 7.2907e-4, 6.7611e-4,
01258
                    6.2758e-4, 5.8308e-4, 5.4223e-4, 5.0473e-4, 4.7027e-4, 4.3859e-4,
                   4.0946e-4, 3.8265e-4, 3.5798e-4, 3.3526e-4, 3.1436e-4, 2.9511e-4, 2.7739e-4, 2.6109e-4, 2.4609e-4, 2.3229e-4, 2.1961e-4, 2.0797e-4,
01259
01260
                    1.9729e-4, 1.875e-4, 1.7855e-4, 1.7038e-4, 1.6294e-4, 1.5619e-4
01261
                    1.5007e-4, 1.4456e-4, 1.3961e-4, 1.3521e-4, 1.3131e-4, 1.2789e-4,
01262
01263
                    1.2494e-4, 1.2242e-4, 1.2032e-4, 1.1863e-4, 1.1733e-4, 1.1641e-4,
01264
                    1.1585e-4, 1.1565e-4, 1.158e-4, 1.1629e-4, 1.1712e-4, 1.1827e-4,
                    1.1976e-4, 1.2158e-4, 1.2373e-4, 1.262e-4, 1.2901e-4, 1.3214e-4,
01265
                    1.3562e-4, 1.3944e-4, 1.4361e-4, 1.4814e-4, 1.5303e-4, 1.5829e-4,
01266
                    1.6394e-4, 1.6999e-4, 1.7644e-4, 1.8332e-4, 1.9063e-4, 1.984e-4,
                   2.0663e-4, 2.1536e-4, 2.246e-4, 2.3436e-4, 2.4468e-4, 2.5558e-4, 2.6708e-4, 2.7921e-4, 2.92e-4, 3.0548e-4, 3.1968e-4, 3.3464e-4,
01268
01269
01270
                   3.5039e-4, 3.6698e-4, 3.8443e-4, 4.0281e-4, 4.2214e-4, 4.4248e-4,
                   4.6389e-4, 4.864e-4, 5.1009e-4, 5.3501e-4, 5.6123e-4, 5.888e-4, 6.1781e-4, 6.4833e-4, 6.8043e-4, 7.142e-4, 7.4973e-4, 7.8711e-4,
01271
```

```
8.2644e-4, 8.6783e-4, 9.1137e-4, 9.5721e-4, .0010054, .0010562,
               .0011096, .0011659, .0012251, .0012875, .0013532, .0014224, .0014953, .001572, .0016529, .0017381, .0018279, .0019226,
01274
01275
01276
                .0020224, .0021277, .0022386, .0023557, .0024792, .0026095,
01277
                .002747, .0028921, .0030453, .0032071, .003378, .0035586,
               .0037494, .003951, .0041642, .0043897, .0046282, .0048805, .0051476, .0054304, .00573, .0060473, .0063837, .0067404, .0071188, .0075203, .0079466, .0083994, .0088806, .0093922,
01280
               .0099366, .010516, .011134, .011792, .012494, .013244, .014046, .014898, .015808, .016781, .017822, .018929, .020108, .02138, .022729, .02419, .02576, .027412, .029233, .031198, .033301, .035594, .038092, .040767, .04372, .046918, .050246, .053974, .058009, .061976, .066586, .071537, .076209, .081856, .087998
01281
01282
01283
01284
                .058009, .061976, .066586, .071537, .076209, .081856,
01285
                .093821, .10113, .10913, .11731, .12724, .13821, .15025, .1639
01286
01287
                .17807, .19472, .21356, .23496, .25758, .28387, .31389, .34104,
               .37469, .40989, .43309, .46845, .5042, .5023, .52981, .55275, .51075, .51976, .52457, .44779, .44721, .4503, .4243, .45244,
01288
01289
               .49491, .55399, .39021, .24802, .2501, .2618, .27475, .28879, .31317, .33643, .36257, .4018, .43275, .46525, .53333, .56599, .60557, .70142, .74194, .77736, .88567, .91182, .93294, .98407,
01290
01292
               .98772, .99176, .9995, 1.2405, 1.3602, 1.338, 1.3255, 1.3267, 1.3404, 1.3634, 1.3967, 1.4407, 1.4961, 1.5603, 1.6328, 1.7153, 1.8094, 1.9091, 2.018, 2.1367, 2.264, 2.4035, 2.5562, 2.7179,
01293
01294
01295
               2.9017, 3.1052, 3.3304, 3.5731, 3.8488, 4.1553, 4.4769, 4.7818, 5.1711, 5.5204, 5.9516, 6.4097, 6.8899, 7.1118, 7.5469, 7.9735,
01296
               7.9511, 8.3014, 8.6418, 8.4757, 8.8256, 9.2294, 9.6923, 10.033, 10.842, 11.851, 11.78, 8.8435, 9.1381, 9.5956, 10.076, 10.629,
01299
01300
               11.22, 11.883, 12.69, 13.163, 13.974, 14.846, 16.027, 17.053,
01301
               18.148, 19.715, 20.907, 22.163, 23.956, 25.235, 26.566, 27.94
               29.576, 30.956, 32.432, 35.337, 39.911, 41.128, 42.625, 44.386, 46.369, 48.619, 51.031, 53.674, 56.825, 59.921, 63.286, 66.929,
01302
01303
01304
                70.859, 75.081, 79.618, 84.513, 89.739, 95.335, 101.35, 107.76,
                114.63, 121.98, 129.87, 138.3, 147.34, 157.04, 167.56, 178.67,
01305
01306
                190.61, 203.43, 217.19, 231.99, 247.88, 264.98, 283.37, 303.17
01307
                324.49, 347.47, 372.25, 398.98, 427.85, 459.06, 492.8, 529.31,
               568.89, 611.79, 658.35, 708.91, 763.87, 823.65, 888.72, 959.58,
01308
               1036.8, 1121.8, 1213.9, 1314.3, 1423.8, 1543., 1672.8, 1813.4, 1966.1, 2131.4, 2309.5, 2499.3, 2705., 2925.7, 3161.6, 3411.3,
01309
01311
                3611.5, 3889.2, 4191.1, 4519.3, 4877.9, 5272.9, 5712.9, 6142.7,
                6719.6, 7385., 8145., 8977.7, 9831.9, 10827., 11934., 13063.,
01312
01313
               14434., 15878., 17591., 19435., 21510., 23835., 26835., 29740.,
                32878., 36305., 39830., 43273., 46931., 50499., 49586., 51598.,
01314
               53429., 54619., 55081., 55102., 54485., 53487., 52042., 42689., 42607., 44020., 47994., 54169., 53916., 55808., 56642., 46049.,
01315
               44243., 32929., 30658., 21963., 20835., 15962., 13679., 17652., 19680., 22388., 25625., 29184., 32520., 35720., 38414., 40523.,
01318
01319
                49228., 48173., 45678., 41768., 37600., 41313., 42654., 44465.,
01320
               55736., 56630., 65409., 63308., 66572., 61845., 60379., 56777.,
               55736., 56630., 65409., 65308., 66572., 61645., 60579., 56777., 51920., 46601., 41367., 36529., 32219., 28470., 25192., 22362., 19907., 17772., 15907., 14273., 12835., 11567., 10445., 9450.2, 8565.1, 7776., 7070.8, 6439.2, 5872.3, 5362.4, 4903., 4488.3, 4113.4, 3773.8, 3465.8, 3186.1, 2931.7, 2700.1, 2488.8, 2296.,
01321
01322
01324
01325
               2119.8, 1958.6, 1810.9, 1675.6, 1551.4, 1437.3, 1332.4, 1236.
               1147.2, 1065.3, 989.86, 920.22, 855.91, 796.48, 741.53, 690.69, 643.62, 600.02, 559.6, 522.13, 487.35, 455.06, 425.08, 397.21, 371.3, 347.2, 324.78, 303.9, 284.46, 266.34, 249.45, 233.7, 219.01, 205.3, 192.5, 180.55, 169.38, 158.95, 149.2, 140.07,
01326
01327
01328
                131.54, 123.56, 116.09, 109.09, 102.54, 96.405, 90.655, 85.266,
01330
                80.213, 75.475, 71.031, 66.861, 62.948, 59.275, 55.827, 52.587,
01331
01332
                49.544, 46.686, 43.998, 41.473, 39.099, 36.867, 34.768, 32.795,
               30.939, 29.192, 27.546, 25.998, 24.539, 23.164, 21.869, 20.65, 19.501, 18.419, 17.399, 16.438, 15.532, 14.678, 13.874, 13.115,
01333
01334
                12.4, 11.726, 11.088, 10.488, 9.921, 9.3846, 8.8784, 8.3996,
               7.9469, 7.5197, 7.1174, 6.738, 6.379, 6.0409, 5.7213, 5.419, 5.1327, 4.8611, 4.6046, 4.3617, 4.1316, 3.9138, 3.7077, 3.5125,
01337
01338
               3.3281, 3.1536, 2.9885, 2.8323, 2.6846, 2.5447, 2.4124, 2.2871,
01339
               2.1686, 2.0564, 1.9501, 1.8495, 1.7543, 1.6641, 1.5787, 1.4978, 1.4212, 1.3486, 1.2799, 1.2147, 1.1529, 1.0943, 1.0388, .98602,
01340
               .93596, .8886, .84352, .80078, .76029, .722, .68585, .65161, .61901, .58808, .55854, .53044, .5039, .47853, .45459, .43173,
01341
                .41008, .38965, .37021, .35186, .33444, .31797, .30234, .28758,
01343
01344
                .2736, .26036, .24764, .2357, .22431, .21342, .20295, .19288,
               18334, 17444, 166, 15815, 15072, 14348, 13674, 13015, 12399, 11807, 11231, 10689, 10164, 096696, 091955, 087476, 083183, 079113, 075229, 071536, 068026, 064698, 06154, 058544, 055699, 052997, 050431, 047993, 045676, 043475, 041382, 039392, 037501, 035702, 033991, 032364,
01345
01346
01347
01348
01349
01350
                .030817, .029345, .027945, .026613, .025345, .024139, .022991,
01351
                .021899, .02086, .019871, .018929, .018033, .01718, .016368,
                .015595, .014859, .014158, .013491, .012856, .012251, .011675,
01352
               .011126, .010604, .010107, .0096331, .009182, .0087523, .0083431, .0079533, .0075821, .0072284, .0068915, .0065706, .0062649,
01353
               .0059737, .0056963, .005432, .0051802, .0049404, .0047118,
01355
                .0044941, .0042867, .0040891, .0039009, .0037216, .0035507
01356
01357
                .003388, .0032329, .0030852, .0029445, .0028105, .0026829,
               .0025613, .0024455, .0023353, .0022303, .0021304, .0020353, .0019448, .0018587, .0017767, .0016988, .0016247, .0015543,
01358
01359
```

```
.0014874, .0014238, .0013635, .0013062, .0012519, .0012005,
                   .0011517, .0011057, .00110621, .001021, 9.8233e-4, 9.4589e-4, 9.1167e-4, 8.7961e-4, 8.4964e-4, 8.2173e-4, 7.9582e-4, 7.7189e-4, 7.499e-4, 7.2983e-4, 7.1167e-4, 6.9542e-4, 6.8108e-4, 6.6866e-4,
01361
01362
01363
                   6.5819e-4, 6.4971e-4, 6.4328e-4, 6.3895e-4, 6.3681e-4, 6.3697e-4, 6.3956e-4, 6.4472e-4, 6.5266e-4, 6.6359e-4, 6.778e-4, 6.9563e-4,
01364
01365
                   7.1749e-4, 7.4392e-4, 7.7556e-4, 8.1028e-4, 8.4994e-4, 8.8709e-4,
                   9.3413e-4, 9.6953e-4, .0010202, .0010738, .0010976, .0011507,
01367
                  9.3415e-4, 9.6933e-4, .0010202, .0010738, .0010976, .001150
.0011686, .0012264, .001291, .0013346, .0014246, .0015293,
.0016359, .0017824, .0019255, .0020854, .002247, .0024148,
.0026199, .0027523, .0029704, .0030702, .0033047, .0035013,
.0037576, .0040275, .0043089, .0046927, .0049307, .0053486,
.0053809, .0056699, .0059325, .0055488, .005634, .0056392,
.004946, .0048855, .0048208, .0044386, .0045498, .0046377,
01368
01369
01370
01371
01372
01373
01374
                   .0048939, .0052396, .0057324, .0060859, .0066906, .0071148,
                  .0077224, .0082687, .008769, .0084471, .008572, .0087729, .008775, .0090742, .0080704, .0080288, .0085747, .0086087, .0086408, .0088752, .0089381, .0089757, .0093532, .0092824, .0092566, .0092645, .0092735, .009342, .0095806, .0097991,
01375
01376
01377
                   .010213, .010611, .011129, .011756, .013237, .01412, .015034,
01379
                  .015936, .01682, .018597, .019315, .019995, .020658, .021289, .022363, .022996, .023716, .024512, .025434, .026067, .027118,
01380
01381
                   .028396, .029865, .031442, .033253, .03525, .037296, .039701,
01382
                  .042356, .045154, .048059, .051294, .054893, .058636, .061407, .065172, .068974, .072676, .073379, .076547, .079556, .079134, .082308, .085739, .090192, .09359, .099599, .10669, .11496,
01383
01384
01385
                   .1244, .13512, .14752, .14494, .15647, .1668, .17863, .19029,
01386
01387
                   .20124, .20254, .21179, .21982, .21625, .22364, .23405, .23382,
01388
                   .2434, .25708, .26406, .27621, .28909, .30395, .31717, .33271,
                   .3496, .36765, .38774, .40949, .446, .46985, .49846, .5287, .562, .59841, .64598, .68834, .7327, .78978, .8373, .88708, .94744, 1.0006, 1.0574, 1.1215, 1.1856, 1.2546, 1.3292, 1.4107, 1.4974,
01389
01390
01391
                   1.5913, 1.6931, 1.8028, 1.9212, 2.0492, 2.1874, 2.3365, 2.4978, 2.6718, 2.8588, 3.062, 3.2818, 3.5188, 3.7752, 4.0527, 4.3542,
01392
01393
                  4.6782, 5.0312, 5.4123, 5.8246, 6.2639, 6.7435, 7.2636, 7.8064, 8.4091, 9.0696, 9.7677, 10.548, 11.4, 12.309, 13.324, 14.284, 15.445, 16.687, 18.019, 19.403, 20.847, 22.366, 23.925, 25.537, 27.213, 28.069, 29.864, 31.829, 33.988, 35.856, 38.829, 42.321,
01394
01395
01396
01398
                   46.319, 50.606, 55.126, 59.126, 64.162, 68.708, 74.615, 81.176,
                   87.739, 95.494, 103.83, 113.38, 123.99, 135.8, 148.7, 162.58,
01399
                  176.32, 192.6, 211.47, 232.7, 252.64, 277.41, 305.38, 333.44, 366.42, 402.66, 442.14, 484.53, 526.42, 568.15, 558.78, 582.6, 600.98, 613.94, 619.44, 618.24, 609.84, 595.96, 484.86, 475.59, 478.49, 501.56, 552.19, 628.44, 630.39, 658.92, 671.96, 562.7,
01400
01401
01402
                   545.88, 423.43, 400.14, 306.59, 294.13, 246.8, 226.51, 278.21,
                  314.39, 347.22, 389.13, 433.16, 477.48, 521.67, 560.54, 683.6, 696.37, 695.91, 683.1, 658.24, 634.89, 698.85, 742.87, 796.66, 954.49, 1009.5, 1150.5, 1179.1, 1267.9, 1272.4, 1312.7, 1330.4, 1331.6, 1315.8, 1308.3, 1293.3, 1274.6, 1249.5, 1213.2, 1172.1, 1124.4, 930.33, 893.36, 871.27, 883.54, 940.76, 1036., 1025.6,
01405
01406
01407
01408
01409
                   1053.1, 914.51, 894.15, 865.03, 670.63, 508.41, 475.15, 370.85,
                  1053.1, 914.51, 894.15, 865.03, 670.63, 508.41, 475.15, 370.85, 361.06, 319.38, 312.75, 331.87, 367.13, 415., 467.94, 525.49, 578.41, 624.66, 794.82, 796.97, 780.29, 736.49, 670.18, 603.75, 659.67, 679.8, 857.12, 884.05, 900.65, 1046.1, 1141.9, 1083., 1089.2, 1e3, 947.08, 872.31, 787.91, 704.75, 624.93, 553.68, 489.91, 434.21, 385.64, 343.3, 306.42, 274.18, 245.94, 221.11, 199.23, 179.88, 162.73, 147.48, 133.88, 121.73, 110.86, 101.1, 102.23, 24.41, 77.73.10, 704.75, 604.54, 244.84, 50.500
01411
01412
01413
01414
01415
                   92.323, 84.417, 77.281, 70.831, 64.991, 59.694, 54.884, 50.509,
01417
                  46.526, 42.893, 39.58, 36.549, 33.776, 31.236, 28.907, 26.77, 24.805, 23., 21.339, 19.81, 18.404, 17.105, 15.909, 14.801,
01418
01419
                   13.778, 12.83, 11.954, 11.142, 10.389, 9.691, 9.0434, 8.4423, 7.8842, 7.3657, 6.8838, 6.4357, 6.0189, 5.6308, 5.2696, 4.9332, 4.6198, 4.3277, 4.0553, 3.8012, 3.5639, 3.3424, 3.1355, 2.9422,
01420
01421
                  2.7614, 2.5924, 2.4343, 2.2864, 2.148, 2.0184, 1.8971, 1.7835, 1.677, 1.5773, 1.4838, 1.3961, 1.3139, 1.2369, 1.1645, 1.0966,
01424
01425
                   1.0329, .97309, .91686, .86406, .81439, .76767, .72381, .68252,
                   .64359, .60695, .57247, .54008, .50957, .48092, .45401, .42862, .40465, .38202, .36072, .34052, .3216, .30386, .28711, .27135, .25651, .24252, .2293, .21689, .20517, .19416, .18381, .17396,
01426
01427
01428
                   .16469
01430
01431
              static double co2230[2001] = { 2.743e-5, 2.8815e-5, 3.027e-5, 3.1798e-5, }
01432
                  3.3405e-5, 3.5094e-5, 3.6869e-5, 3.8734e-5, 4.0694e-5, 4.2754e-5, 4.492e-5, 4.7196e-5, 4.9588e-5, 5.2103e-5, 5.4747e-5, 5.7525e-5,
01433
01434
                   6.0446e-5, 6.3516e-5, 6.6744e-5, 7.0137e-5, 7.3704e-5, 7.7455e-5,
01435
                   8.1397e-5, 8.5543e-5, 8.9901e-5, 9.4484e-5, 9.9302e-5, 1.0437e-4,
01436
01437
                   1.097e-4, 1.153e-4, 1.2119e-4, 1.2738e-4, 1.3389e-4, 1.4074e-4,
                   1.4795e-4, 1.5552e-4, 1.6349e-4, 1.7187e-4, 1.8068e-4, 1.8995e-4, 1.997e-4, 2.0996e-4, 2.2075e-4, 2.321e-4, 2.4403e-4, 2.5659e-4, 2.698e-4, 2.837e-4, 2.9832e-4, 3.137e-4, 3.2988e-4, 3.4691e-4,
01438
01439
01440
                   3.6483e-4, 3.8368e-4, 4.0351e-4, 4.2439e-4, 4.4635e-4, 4.6947e-4,
                   4.9379e-4, 5.1939e-4, 5.4633e-4, 5.7468e-4, 6.0452e-4, 6.3593e-4,
01442
01443
                   6.69e-4, 7.038e-4, 7.4043e-4, 7.79e-4, 8.1959e-4, 8.6233e-4,
01444
                   9.0732e-4, 9.5469e-4, .0010046, .0010571, .0011124, .0011706,
                  .0012319, .0012964, .0013644, .001436, .0015114, .0015908, .0016745, .0017625, .0018553, .0019531, .002056, .0021645,
01445
```

```
.0022788, .0023992, .002526, .0026596, .0028004, .0029488,
                 .0031052, .0032699, .0034436, .0036265, .0038194, .0040227, .0042369, .0044628, .0047008, .0049518, .0052164, .0054953,
01448
01449
01450
                  .0057894, .0060995, .0064265, .0067713, .007135, .0075184,
01451
                  .0079228, .0083494,
                                                     .0087993. .0092738.
                                                                                         .0097745.
                                                                                                           .010303.
                 .001086, .011448, .012068, .012722, .013413, .014142, .014911, .015723, .01658, .017484, .018439, .019447, .020511, .021635,
01452
                  .022821, .024074, .025397, .026794, .02827, .029829,
01454
01455
                  .033215, .035052, .036994, .039045, .041213, .043504, .045926,
                 .048485, .05119, .05405, .057074, .060271, .063651, .067225, .071006, .075004, .079233, .083708, .088441, .093449, .098749, .10436, .11029, .11657, .12322, .13026, .13772, .14561, .15397, .16282, .1722, .18214, .19266, .20381, .21563, .22816, .24143, .2555, .27043, .28625, .30303, .32082, .3397, .35972, .38097,
01456
01457
01458
01459
01460
01461
                  .40352, .42746, .45286, .47983, .50847, .53888, .57119, .6055
                 . 164196, 6807, 72187, 76564, 81217, 86165, 91427, 97025, 1.0298, 1.0932, 1.1606, 1.2324, 1.3088, 1.3902, 1.477, 1.5693, 1.6678, 1.7727, 1.8845, 2.0038, 2.131, 2.2666, 2.4114, 2.5659, 2.7309, 2.907, 3.0951, 3.2961, 3.5109, 3.7405, 3.986, 4.2485,
01462
01463
01464
                  4.5293, 4.8299, 5.1516, 5.4961, 5.8651, 6.2605, 6.6842, 7.1385,
                 4.3293, 4.6293, 3.1316, 3.4961, 3.8631, 6.2603, 6.8642, 7.1363, 7.6256, 8.1481, 8.7089, 9.3109, 9.9573, 10.652, 11.398, 12.2, 13.063, 13.992, 14.99, 16.064, 17.222, 18.469, 19.813, 21.263, 22.828, 24.516, 26.34, 28.31, 30.437, 32.738, 35.226, 37.914, 40.824, 43.974, 47.377, 51.061, 55.011, 59.299, 63.961, 69.013, 74.492, 80.444, 86.919, 93.836, 101.23, 109.25, 117.98, 127.47,
01467
01468
01469
01470
01471
                  137.81, 149.07, 161.35, 174.75, 189.42, 205.49, 223.02, 242.26,
                  263.45, 286.75, 311.94, 340.01, 370.86, 404.92, 440.44, 480.27,
01473
                 525.17, 574.71, 626.22, 686.8, 754.38, 827.07, 913.38, 1011.7,
01474
                 121.5, 1161.6, 1289.5, 1432.2, 1595.4, 1777., 1983.3, 2216.1, 2485.7, 2788.3, 3101.5, 3481., 3902.1, 4257.1, 4740., 5272.8, 5457.9, 5946.2, 6505.3, 6668.4, 7302.4, 8061.6, 9015.8, 9908.3, 11613., 13956., 3249.6, 3243., 2901.5, 2841.3, 2729.6, 2558.2, 1797.8, 1583.2, 1386., 1233.5, 787.74, 701.46, 761.66, 767.21,
01475
01476
01477
01479
01480
                  722.83, 1180.6, 1332.1, 1461.6, 2032.9, 2166., 2255.9, 2294.7,
                 2587.2, 2396.5, 2122.4, 12553., 10784., 9832.5, 8827.3, 8029.1, 7377.9, 7347.1, 6783.8, 6239.1, 5721.1, 5503., 4975.1, 4477.8, 4021.3, 3676.8, 3275.3, 2914.9, 2597.4, 2328.2, 2075.4, 1857.6, 1663.6, 1493.3, 1343.8, 1213.3, 1095.6, 1066.5, 958.91, 865.15,
01481
01482
01483
01485
                  783.31, 714.35, 650.77, 593.98, 546.2, 499.9, 457.87, 421.75,
                 387.61, 355.25, 326.62, 299.7, 275.21, 253.17, 232.83, 214.31, 197.5, 182.08, 167.98, 155.12, 143.32, 132.5, 122.58, 113.48,
01486
01487
                 105.11, 97.415, 90.182, 83.463, 77.281, 71.587, 66.341, 61.493, 57.014, 53.062, 49.21, 45.663, 42.38, 39.348, 36.547, 33.967, 31.573, 29.357, 27.314, 25.415, 23.658, 22.03, 20.524, 19.125,
01488
01489
01490
                 17.829, 16.627, 15.511, 14.476, 13.514, 12.618, 11.786, 11.013, 10.294, 9.6246, 9.0018, 8.4218, 7.8816, 7.3783, 6.9092, 6.4719,
01492
01493
                  6.0641, 5.6838, 5.3289, 4.998, 4.6893, 4.4014, 4.1325, 3.8813,
01494
                 3.6469, 3.4283, 3.2241, 3.035, 2.8576, 2.6922, 2.5348, 2.3896,
                 2.2535, 2.1258, 2.0059, 1.8929, 1.7862, 1.6854, 1.5898, 1.4992,
01495
                 1.4017, 1.3218, 1.2479, 1.1809, 1.1215, 1.0693, 1.0116, .96016, .9105, .84859, .80105, .74381, .69982, .65127, .60899, .57843, .54592, .51792, .49336, .47155, .45201, .43426, .41807, .40303,
01496
01498
01499
                  .38876, .3863, .37098, .35492, .33801, .32032, .30341, .29874,
01500
                  .29193, .28689, .29584, .29155, .29826, .29195, .29287, .2904
                 .28199, .27709, .27162, .26622, .26133, .25676, .25235, .23137,
01501
                 .22365, .21519, .20597, .19636, .18699, .16485, .16262, .16643, .17542, .18198, .18631, .16759, .16338, .13505, .1267, .10053,
01502
                 .092554, .074093, .062159, .055523, .054849, .05401,
01504
                 .058982, .07952, .08647, .093244, .099285, .10393, .10661, .12072, .11417, .10396, .093265, .089137, .088909, .10902,
01505
01506
                 .11277, .13625, .13565, .14907, .14167, .1428, .13744, .12768, .11382, .10244, .091686, .08109, .071739, .063616, .056579, .050504, .045251, .040689, .036715, .033237, .030181, .027488,
01507
01508
                 .025107, .022998, .021125, .01946, .017979, .016661, .015489, .014448, .013526, .012712, .011998, .011375, .010839, .010384
01511
                                                                                                                   .010384.
01512
                  .010007, .0097053, .0094783, .0093257, .0092489, .0092504,
                 .0093346, .0095077, .0097676, .01012, .01058, .011157, .011844, .012672, .013665, .014766, .015999, .017509, .018972, .020444, .022311, .023742, .0249, .025599, .026981, .026462, .025143,
01513
01514
01515
                 .025066, .022814, .020458, .020026, .019142, .020189, .022371,
                 .024163, .023728, .02199, .019506, .018591, .015576, .012784,
01517
01518
                 .011744, .0094777, .0079148, .0070652, .006986, .0071758,
01519
                  .008086, .0098025, .01087, .013609, .016764, .018137, .021061,
                  .023498, .023576, .023965, .022828, .021519, .021283, .023364,
01520
                 .026457, .029782, .030856, .033486, .035515, .035543, .036558, .037198, .037472, .037045, .037284, .03777, .038085, .038366, .038526, .038282, .038915, .037697, .035667, .032941, .031959,
01521
01523
                 .028692, .025918, .024596, .025592, .027873, .028935, .02984, .028148, .025305, .021912, .020454, .016732, .013357, .01205,
01524
01525
                  .009731, .0079881, .0077704, .0074387, .0083895, .0096776,
01526
                 .010326, .01293, .015955, .019247, .020145, .02267, .024231, .024184, .022131, .019784, .01955, .01971, .022119, .025116,
01527
                 .027978, .028107, .029808, .030701, .029164, .028551, .027286, .024946, .023259, .020982, .019221, .017471, .015643, .014074,
01529
01530
                 .01261, .011301, .010116, .0090582, .0081036, .0072542, .0065034, .0058436, .0052571, .0047321, .0042697, .0038607, .0034977, .0031747, .0028864, .0026284, .002397, .002189, .0020017,
01531
01532
01533
```

```
.0018326, .0016798, .0015414, .0014159, .0013019, .0011983,
                .0011039, .0010177, 9.391e-4, 8.6717e-4, 8.0131e-4, 7.4093e-4,
01535
01536
               6.8553e-4, 6.3464e-4, 5.8787e-4, 5.4487e-4, 5.0533e-4, 4.69e-4,
               4.3556e-4, 4.0474e-4, 3.7629e-4, 3.5e-4, 3.2569e-4, 3.032e-4,
01537
               2.8239e-4, 2.6314e-4, 2.4535e-4, 2.2891e-4, 2.1374e-4, 1.9975e-4,
01538
                1.8685e-4, 1.7498e-4, 1.6406e-4, 1.5401e-4, 1.4479e-4, 1.3633e-4,
                1.2858e-4, 1.2148e-4, 1.1499e-4, 1.0907e-4, 1.0369e-4, 9.8791e-5,
                9.4359e-5, 9.0359e-5, 8.6766e-5, 8.3555e-5, 8.0703e-5, 7.8192e-5,
01541
01542
               7.6003e-5, 7.4119e-5, 7.2528e-5, 7.1216e-5, 7.0171e-5, 6.9385e-5,
               6.8848e-5, 6.8554e-5, 6.8496e-5, 6.8669e-5, 6.9069e-5, 6.9694e-5, 7.054e-5, 7.1608e-5, 7.2896e-5, 7.4406e-5, 7.6139e-5, 7.8097e-5, 8.0283e-5, 8.2702e-5, 8.5357e-5, 8.8255e-5, 9.1402e-5, 9.4806e-5,
01543
01544
01545
               9.8473e-5, 1.0241e-4, 1.0664e-4, 1.1115e-4, 1.1598e-4, 1.2112e-4,
               1.2659e-4, 1.3241e-4, 1.3859e-4, 1.4515e-4, 1.521e-4, 1.5947e-4,
01547
01548
               1.6728e-4, 1.7555e-4, 1.8429e-4, 1.9355e-4, 2.0334e-4, 2.1369e-4,
01549
               2.2463e-4, 2.3619e-4, 2.4841e-4, 2.6132e-4, 2.7497e-4, 2.8938e-4,
               3.0462e-4, 3.2071e-4, 3.3771e-4, 3.5567e-4, 3.7465e-4, 3.947e-4,
01550
               4.1588e-4, 4.3828e-4, 4.6194e-4, 4.8695e-4, 5.1338e-4, 5.4133e-4, 5.7087e-4, 6.0211e-4, 6.3515e-4, 6.701e-4, 7.0706e-4, 7.4617e-4,
01551
               7.8756e-4, 8.3136e-4, 8.7772e-4, 9.2681e-4, 9.788e-4,
               .0010922, .001154, .0012195, .0012889, .0013626, .0014407, .0015235, .0016114, .0017048, .0018038, .001909, .0020207,
01554
01555
                .0021395, .0022657, .0023998, .0025426, .0026944, .002856,
01556
              .0032181, .0032114, .0034068, .003615, .0038371, .004074, .004327, .0045971, .0048857, .0051942, .0055239, .0058766, .0062538, .0066573, .0070891, .007551, .0080455, .0085747, .0091412, .0097481, .010397, .011092, .011837, .012638, .013495,
01557
01560
               .014415, .01541, .016475, .017621, .018857, .020175, .02162, .023185, .024876, .02672, .028732, .030916, .033319, .035939,
01561
01562
               .038736, .041847, .04524, .048715, .052678, .056977, .061203, .066184, .07164, .076952, .083477, .090674, .098049, .10697,
01563
01564
               .1169, .1277, .14011, .15323, .1684, .18601, .20631, .20831, .25417, .28407, .31405, .34957, .38823, .41923, .46026, .50409, .51227, .54805, .57976, .53818, .55056, .557, .46741, .46403,
01566
01567
01568
                .4636, .42265, .45166, .49852, .56663, .34306, .17779, .17697
               . 18346, . 19129, . 20014, . 21778, . 23604, . 25649, . 28676, . 31238, . 33856, . 39998, . 4288, . 46568, . 56654, . 60786, . 64473, . 76466, . 7897, . 80778, . 86443, . 85736, . 84798, . 84157, 1.1385, 1.2446, 1.1923, 1.1552, 1.1338, 1.1266, 1.1292, 1.1431, 1.1683, 1.2059,
01569
01570
01572
               1.2521, 1.3069, 1.3712, 1.4471, 1.5275, 1.6165, 1.7145, 1.8189, 1.9359, 2.065, 2.2007, 2.3591, 2.5362, 2.7346, 2.9515, 3.2021,
01573
01574
               3.4851, 3.7935, 4.0694, 4.4463, 4.807, 5.2443, 5.7178, 6.2231, 6.4796, 6.9461, 7.4099, 7.3652, 7.7182, 8.048, 7.7373, 8.0363, 8.3855, 8.8044, 9.0257, 9.8574, 10.948, 10.563, 6.8979, 7.0744,
01575
01576
               7.4121, 7.7663, 8.1768, 8.6243, 9.1437, 9.7847, 10.182, 10.849, 11.572, 12.602, 13.482, 14.431, 15.907, 16.983, 18.11, 19.884,
01579
01580
               21.02, 22.18, 23.355, 24.848, 25.954, 27.13, 30.186, 34.893,
               35.682, 36.755, 38.111, 39.703, 41.58, 43.606, 45.868, 48.573, 51.298, 54.291, 57.559, 61.116, 64.964, 69.124, 73.628, 78.471,
01581
01582
               83.683, 89.307, 95.341, 101.84, 108.83, 116.36, 124.46, 133.18,
01583
               142.57, 152.79, 163.69, 175.43, 188.11, 201.79, 216.55, 232.51,
               249.74, 268.38, 288.54, 310.35, 333.97, 359.55, 387.26, 417.3, 449.88, 485.2, 523.54, 565.14, 610.28, 659.31, 712.56, 770.43,
01585
01586
               833.36, 901.82, 976.36, 1057.6, 1146.8, 1243.8, 1350., 1466.3, 1593.6, 1732.7, 1884.1, 2049.1, 2228.2, 2421.9, 2629.4, 2853.7,
01587
01588
01589
               3094.4, 3351.1, 3622.3, 3829.8, 4123.1, 4438.3, 4777.2, 5144.1,
               5545.4, 5990.5, 6404.5, 6996.8, 7687.6, 8482.9, 9349.4, 10203.,
               11223., 12358., 13493., 14916., 16416., 18236., 20222., 22501.,
01591
01592
               25102., 28358., 31707., 35404., 39538., 43911., 48391., 53193.,
01593
               58028., 58082., 61276., 64193., 66294., 67480., 67921., 67423.,
               66254, 64341, 51737., 51420., 53072., 58145., 66195., 65358., 67377., 67869., 53509., 50553., 35737., 32425., 21704., 19974., 14457., 12142., 16798., 19489., 23049., 27270., 31910., 36457.,
01594
01595
               40877., 44748., 47876., 59793., 58626., 55454., 50337., 44893., 50228., 52216., 54747., 69541., 70455., 81014., 77694., 80533.,
01597
01598
01599
               73953., 70927., 65539., 59002., 52281., 45953., 40292., 35360.,
               31124, 27478., 24346., 21647., 19308., 17271., 15491., 13927., 12550., 11331., 10250., 9288.8, 8431.4, 7664.9, 6978.3, 6361.8, 5807.4, 5307.7, 4856.8, 4449., 4079.8, 3744.9, 3440.8, 3164.2, 2912.3, 2682.7, 2473., 2281.4, 2106., 1945.3, 1797.9, 1662.5,
01600
01601
01602
               1538.1, 1423.6, 1318.1, 1221., 1131.5, 1049., 972.99, 902.87, 838.01, 777.95, 722.2, 670.44, 622.35, 577.68, 536.21, 497.76,
01604
01605
               462.12, 429.13, 398.61, 370.39, 344.29, 320.16, 297.85, 277.2, 258.08, 240.38, 223.97, 208.77, 194.66, 181.58, 169.43, 158.15,
01606
01607
               147.67, 137.92, 128.86, 120.44, 112.6, 105.3, 98.499, 92.166, 86.264, 80.763, 75.632, 70.846, 66.381, 62.213, 58.321, 54.685,
01608
01609
               51.288, 48.114, 45.145, 42.368, 39.772, 37.341, 35.065, 32.937,
01610
01611
               30.943, 29.077, 27.33, 25.693, 24.158, 22.717, 21.367, 20.099,
               18.909, 17.792, 16.744, 15.761, 14.838, 13.971, 13.157, 12.393, 11.676, 11.003, 10.369, 9.775, 9.2165, 8.6902, 8.1963, 7.7314, 7.2923, 6.8794, 6.4898, 6.122, 5.7764, 5.4525, 5.1484, 4.8611,
01612
01613
01614
                4.5918, 4.3379, 4.0982, 3.8716, 3.6567, 3.4545, 3.2634, 3.0828,
               2.9122, 2.7512, 2.5993, 2.4561, 2.3211, 2.1938, 2.0737, 1.9603,
01616
01617
               1.8534, 1.7525, 1.6572, 1.5673, 1.4824, 1.4022, 1.3265, 1.2551,
01618
               1.1876, 1.1239, 1.0637, 1.0069, .9532, .90248, .85454, .80921,
               .76631, .72569, .6872, .65072, .61635, .5836, .55261, .52336, .49581, .46998, .44559, .42236, .40036, .37929, .35924, .34043,
01619
01620
```

```
.32238, .30547, .28931, .27405, .25975, .24616, .23341, .22133,
               .20997, .19924, .18917, .17967, .17075, .16211, .15411, .14646, .13912, .13201, .12509, .11857, .11261, .10698, .10186, .097039
01622
01623
                                                                                                           .097039.
                .092236, .087844, .083443, .07938, .075452, .071564, .067931,
01624
               .064389, .061078, .057901, .054921, .052061, .049364, .046789, .04435, .042044, .039866, .037808, .035863, .034023, .032282, .030634, .029073, .027595, .026194, .024866, .023608, .022415, .021283, .02021, .019193, .018228, .017312, .016443, .015619,
01625
01626
01629
               .014837, .014094, .01339, .012721, .012086, .011483, .010911, .010368, .009852, .0093623, .0088972, .0084556, .0080362, .0076379, .0072596, .0069003, .006559, .0062349, .0059269,
01630
01631
               .0056344, .0053565, .0050925, .0048417, .0046034, .004377, .0041618, .0039575, .0037633, .0035788, .0034034, .0032368,
01632
01633
               .0030785, .002928, .0027851, .0026492, .0025201, .0023975,
01634
01635
               .0022809, .0021701, .0020649, .0019649, .0018699, .0017796,
               .0016938, .0016122, .0015348, .0014612, .0013913, .001325, .0012619, .0012021, .0011452, .0010913, .0010401, 9.9149e-4, 9.454e-4, 9.0169e-4, 8.6024e-4, 8.2097e-4, 7.8377e-4, 7.4854e-4,
01636
01637
01638
                7.1522e-4, 6.8371e-4, 6.5393e-4, 6.2582e-4, 5.9932e-4, 5.7435e-4,
01639
                5.5087e-4, 5.2882e-4, 5.0814e-4, 4.8881e-4, 4.7076e-4, 4.5398e-4,
                4.3843e-4, 4.2407e-4, 4.109e-4, 3.9888e-4, 3.88e-4, 3.7826e-4
01641
01642
               3.6963e-4, 3.6213e-4, 3.5575e-4, 3.505e-4, 3.464e-4, 3.4346e-4,
               3.4173e-4, 3.4125e-4, 3.4206e-4, 3.4424e-4, 3.4787e-4, 3.5303e-4,
01643
               3.5986e-4, 3.6847e-4, 3.7903e-4, 3.9174e-4, 4.0681e-4, 4.2455e-4, 4.4527e-4, 4.6942e-4, 4.9637e-4, 5.2698e-4, 5.5808e-4, 5.9514e-4,
01644
01645
               6.2757e-4, 6.689e-4, 7.1298e-4, 7.3955e-4, 7.8403e-4, 8.0449e-4, 8.5131e-4, 9.0256e-4, 9.3692e-4, .0010051, .0010846, .0011678,
01646
01647
01648
               .001282, .0014016, .0015355, .0016764, .0018272, .0020055,
01649
                .0021455, .0023421, .0024615, .0026786, .0028787, .0031259,
               .0034046, .0036985, .0040917, .0043902, .0048349, .0049531, .0052989, .0056148, .0052452, .0053357, .005333, .0045069,
01650
01651
               .0043851, .004253, .003738, .0038084, .0039013, .0041505, .0045372, .0050569, .0054507, .0061267, .0066122, .0072449, .0078012, .0082651, .0076538, .0076573, .0076806, .0075227,
01653
01654
               .0076269, .0063758, .006254, .0067749, .0067909, .0068231, .0072143, .0072762, .0072954, .007679, .0075107, .0073658,
01655
01656
               .0072441, .0071074, .0070378, .007176, .0072472, .0075844, .0079291, .008412, .0090165, .010688, .011535, .012375, .013166, .013895, .015567, .016011, .016392, .016737, .017043, .017731,
01657
               .018031, .018419, .018877, .019474, .019868, .020604, .021538
.022653, .023869, .025288, .026879, .028547, .030524, .03274,
01660
01661
               .035132, .03769, .040567, .043793, .047188, .049962, .053542, .057205, .060776, .061489, .064419, .067124, .065945, .068487, .071209, .074783, .077039, .082444, .08902, .09692, .10617,
01662
01663
01664
               .11687, .12952, .12362, .13498, .14412, .15492, .16519, .1744, .17096, .17714, .18208, .17363, .17813, .18564, .18295, .19045,
01666
01667
                .20252, .20815, .21844, .22929, .24229, .25321, .26588, .2797,
01668
               .29465, .31136, .32961, .36529, .38486, .41027, .43694, .4667,
                .49943, .54542, .58348, .62303, .67633, .71755, .76054, .81371, .85934, .90841, .96438, 1.0207, 1.0821, 1.1491, 1.2226, 1.3018, 1.388, 1.4818, 1.5835, 1.6939, 1.8137, 1.9435, 2.0843, 2.237,
01669
01670
                2.4026, 2.5818, 2.7767, 2.9885, 3.2182, 3.4679, 3.7391, 4.0349,
01672
01673
                4.3554, 4.7053, 5.0849, 5.4986, 5.9436, 6.4294, 6.9598, 7.5203,
               8.143, 8.8253, 9.5568, 10.371, 11.267, 12.233, 13.31, 14.357, 15.598, 16.93, 18.358, 19.849, 21.408, 23.04, 24.706, 26.409,
01674
01675
               28.153, 28.795, 30.549, 32.43, 34.49, 36.027, 38.955, 42.465, 46.565, 50.875, 55.378, 59.002, 63.882, 67.949, 73.693, 80.095
01676
                86.403, 94.264, 102.65, 112.37, 123.3, 135.54, 149.14, 163.83,
01678
                179.17, 196.89, 217.91, 240.94, 264.13, 292.39, 324.83, 358.21,
01679
01680
               397.16, 440.5, 488.6, 541.04, 595.3, 650.43, 652.03, 688.74,
               719.47, 743.54, 757.68, 762.35, 756.43, 741.42, 595.43, 580.97, 580.83, 605.68, 667.88, 764.49, 759.93, 789.12, 798.17, 645.66, 615.65, 455.05, 421.09, 306.45, 289.14, 235.7, 215.52, 274.57,
01681
01682
                316.53, 357.73, 409.89, 465.06, 521.84, 579.02, 630.64, 794.46,
01685
               813., 813.56, 796.25, 761.57, 727.97, 812.14, 866.75, 932.5,
01686
               1132.8, 1194.8, 1362.2, 1387.2, 1482.3, 1479.7, 1517.9, 1533.1,
01687
               1534.2, 1523.3, 1522.5, 1515.5, 1505.2, 1486.5, 1454., 1412.,
                1358.8, 1107.8, 1060.9, 1033.5, 1048.2, 1122.4, 1248.9, 1227.1,
01688
                1255.4, 1058.9, 1020.7, 970.59, 715.24, 512.56, 468.47, 349.3,
01689
                338.26, 299.22, 301.26, 332.38, 382.08, 445.49, 515.87, 590.85, 662.3, 726.05, 955.59, 964.11, 945.17, 891.48, 807.11, 720.9,
01691
01692
               803.36, 834.46, 1073.9, 1107.1, 1123.6, 1296., 1393.7, 1303.1,
               1284.3, 1161.8, 1078.8, 976.13, 868.72, 767.4, 674.72, 593.73, 523.12, 462.24, 409.75, 364.34, 325., 290.73, 260.76, 234.46,
01693
01694
               211.28, 190.78, 172.61, 156.44, 142.01, 129.12, 117.57, 107.2, 97.877, 89.47, 81.882, 75.021, 68.807, 63.171, 58.052, 53.396,
01695
                49.155, 45.288, 41.759, 38.531, 35.576, 32.868, 30.384, 28.102,
01697
01698
                26.003, 24.071, 22.293, 20.655, 19.147, 17.756, 16.476, 15.292,
               14.198, 13.183, 12.241, 11.367, 10.554, 9.7989, 9.0978, 8.4475, 7.845, 7.2868, 6.7704, 6.2927, 5.8508, 5.4421, 5.064, 4.714, 4.3902, 4.0902, 3.8121, 3.5543, 3.315, 3.093, 2.8869, 2.6953, 2.5172, 2.3517, 2.1977, 2.0544, 1.9211, 1.7969, 1.6812, 1.5735,
01699
01700
01701
               1.4731, 1.3794, 1.2921, 1.2107, 1.1346, 1.0637, .99744, .93554, .87771, .82368, .77313, .72587, .6816, .64014, .60134, .565,
01703
01704
01705
                .53086, .49883, .46881, .44074, .4144, .38979, .36679, .34513,
               .32474, .30552, .28751, .27045, .25458, .23976, .22584, .21278, .20051, .18899, .17815, .16801, .15846, .14954, .14117, .13328,
01706
01707
```

```
01708
           .12584
01709
01710
01711
         double xw, dw, ew, cw296, cw260, cw230, dt230, dt260, dt296, ctw, ctmpth;
01712
01713
         int iw, iw;
01714
01715
         /* Get CO2 continuum absorption... */
         xw = nu / 2 + 1;
if (xw >= 1 && xw < 2001) {
01716
01717
           iw = (int) xw;
01718
            jw = iw + 1;
01719
           dw = xw - iw;
01720
01721
            ew = 1 - dw;
01722
            cw296 = ew * co2296[iw - 1] + dw * co2296[jw - 1];
           cw260 = ew * co2260[iw - 1] + dw * co2260[jw - 1];

cw230 = ew * co2230[iw - 1] + dw * co2230[jw - 1];
01723
01724
01725
           dt230 = t - 230;
           dt260 = t - 260;
           dt296 = t - 296;
01727
           ctw = dt260 * 5.050505e-4 * dt296 * cw230 - dt230 * 9.259259e-4 * dt296 * cw260 + dt230 * 4.208754e-4 * dt260 * cw296;
01728
01729
            ctmpth = u / GSL_CONST_NUM_AVOGADRO / 1000 * p / P0 * ctw;
01730
01731
         } else
           ctmpth = 0;
01732
         return ctmpth;
01734 }
```

#### 5.13.2.7 double ctmh2o ( double nu, double p, double t, double q, double u )

Compute water vapor continuum (optical depth).

Definition at line 1738 of file jurassic.c.

```
01743
01744
01745
         static double h2o296[2001] = \{ .17, .1695, .172, 
01746
           .1508, .1447, .1344, .1214, .1133, .1009, .09217, .08297,
                                                                                     .06989,
            .06513, .05469, .05056, .04417, .03779, .03484, .02994, .0272,
01747
            .02325, .02063, .01818, .01592, .01405, .01251, .0108, .009647,
01748
            .008424, .007519, .006555, .00588, .005136, .004511, .003989, .003509, .003114, .00274, .002446, .002144, .001895, .001676,
01749
01751
            .001486, .001312, .001164, .001031, 9.129e-4, 8.106e-4, 7.213e-4,
01752
            6.4e-4, 5.687e-4, 5.063e-4, 4.511e-4, 4.029e-4, 3.596e-4,
           3.22e-4, 2.889e-4, 2.597e-4, 2.337e-4, 2.108e-4, 1.907e-4, 1.728e-4, 1.57e-4, 1.43e-4, 1.305e-4, 1.195e-4, 1.097e-4,
01753
01754
01755
            1.009e-4, 9.307e-5, 8.604e-5, 7.971e-5, 7.407e-5, 6.896e-5,
01756
            6.433e-5, 6.013e-5, 5.631e-5, 5.283e-5, 4.963e-5, 4.669e-5,
            4.398e-5, 4.148e-5, 3.917e-5, 3.702e-5, 3.502e-5, 3.316e-5,
01757
01758
            3.142e-5, 2.978e-5, 2.825e-5, 2.681e-5, 2.546e-5, 2.419e-5
01759
            2.299e-5, 2.186e-5, 2.079e-5, 1.979e-5, 1.884e-5, 1.795e-5,
            1.711e-5, 1.633e-5, 1.559e-5, 1.49e-5, 1.426e-5, 1.367e-5,
01760
            1.312e-5, 1.263e-5, 1.218e-5, 1.178e-5, 1.143e-5, 1.112e-5, 1.088e-5, 1.07e-5, 1.057e-5, 1.05e-5, 1.051e-5, 1.059e-5,
01761
01763
            1.076e-5, 1.1e-5, 1.133e-5, 1.18e-5, 1.237e-5, 1.308e-5,
01764
            1.393e-5, 1.483e-5, 1.614e-5, 1.758e-5, 1.93e-5, 2.123e-5
01765
            2.346e-5, 2.647e-5, 2.93e-5, 3.279e-5, 3.745e-5, 4.152e-5,
01766
            4.813e-5, 5.477e-5, 6.203e-5, 7.331e-5, 8.056e-5, 9.882e-5, 1.05e-4, 1.21e-4, 1.341e-4, 1.572e-4, 1.698e-4, 1.968e-4,
01767
            2.175e-4, 2.431e-4, 2.735e-4, 2.867e-4, 3.19e-4, 3.371e-4,
01768
            3.554e-4, 3.726e-4, 3.837e-4, 3.878e-4, 3.864e-4, 3.858e-4,
01770
            3.841e-4, 3.852e-4, 3.815e-4, 3.762e-4, 3.618e-4, 3.579e-4,
            3.45e-4, 3.202e-4, 3.018e-4, 2.785e-4, 2.602e-4, 2.416e-4,
01771
01772
            2.097e-4, 1.939e-4, 1.689e-4, 1.498e-4, 1.308e-4, 1.17e-4,
            1.011e-4, 9.237e-5, 7.909e-5, 7.006e-5, 6.112e-5, 5.401e-5,
01773
            4.914e-5, 4.266e-5, 3.963e-5, 3.316e-5, 3.037e-5, 2.598e-5,
01774
            2.294e-5, 2.066e-5, 1.813e-5, 1.583e-5, 1.423e-5, 1.247e-5, 1.116e-5, 9.76e-6, 8.596e-6, 7.72e-6, 6.825e-6, 6.108e-6,
01776
            5.366e-6, 4.733e-6, 4.229e-6, 3.731e-6, 3.346e-6, 2.972e-6, 2.628e-6, 2.356e-6, 2.102e-6, 1.878e-6, 1.678e-6, 1.507e-6,
01777
01778
            1.348e-6, 1.21e-6, 1.089e-6, 9.806e-7, 8.857e-7, 8.004e-7, 7.261e-7, 6.599e-7, 6.005e-7, 5.479e-7, 5.011e-7, 4.595e-7, 4.219e-7, 3.885e-7, 3.583e-7, 3.314e-7, 3.071e-7, 2.852e-7,
01779
01780
01782
            2.654e-7, 2.474e-7, 2.311e-7, 2.162e-7, 2.026e-7, 1.902e-7,
01783
            1.788e-7, 1.683e-7, 1.587e-7, 1.497e-7, 1.415e-7,
01784
            1.266e-7, 1.2e-7, 1.138e-7, 1.08e-7, 1.027e-7, 9.764e-8,
            9.296e-8, 8.862e-8, 8.458e-8, 8.087e-8, 7.744e-8, 7.429e-8, 7.145e-8, 6.893e-8, 6.664e-8, 6.468e-8, 6.322e-8, 6.162e-8,
01785
01786
01787
            6.07e-8, 5.992e-8, 5.913e-8, 5.841e-8, 5.796e-8, 5.757e-8,
            5.746e-8, 5.731e-8, 5.679e-8, 5.577e-8, 5.671e-8, 5.656e-8,
```

```
5.594e-8, 5.593e-8, 5.602e-8, 5.62e-8, 5.693e-8, 5.725e-8,
              5.858e-8, 6.037e-8, 6.249e-8, 6.535e-8, 6.899e-8, 7.356e-8, 7.918e-8, 8.618e-8, 9.385e-8, 1.039e-7, 1.158e-7, 1.29e-7,
01790
01791
              1.437e-7, 1.65e-7, 1.871e-7, 2.121e-7, 2.427e-7, 2.773e-7, 3.247e-7, 3.677e-7, 4.037e-7, 4.776e-7, 5.101e-7, 6.214e-7, 6.936e-7, 7.581e-7, 8.486e-7, 9.355e-7, 9.942e-7, 1.063e-6,
01792
01793
01794
01795
              1.123e-6, 1.191e-6, 1.215e-6, 1.247e-6, 1.26e-6, 1.271e-6,
              1.284e-6, 1.317e-6, 1.323e-6, 1.349e-6, 1.353e-6, 1.362e-6,
01796
01797
              1.344e-6, 1.329e-6, 1.336e-6, 1.327e-6, 1.325e-6, 1.359e-6,
01798
              1.374e-6, 1.415e-6, 1.462e-6, 1.526e-6, 1.619e-6, 1.735e-6,
              1.863e-6, 2.034e-6, 2.265e-6, 2.482e-6, 2.756e-6, 3.103e-6,
01799
              3.466e-6, 3.832e-6, 4.378e-6, 4.913e-6, 5.651e-6, 6.311e-6,
01800
              7.169e-6, 8.057e-6, 9.253e-6, 1.047e-5, 1.212e-5, 1.36e-5,
01801
              1.569e-5, 1.776e-5, 2.02e-5, 2.281e-5, 2.683e-5, 2.994e-5,
01802
01803
              3.488e-5, 3.896e-5, 4.499e-5, 5.175e-5, 6.035e-5, 6.34e-5
01804
              7.281e-5, 7.923e-5, 8.348e-5, 9.631e-5, 1.044e-4, 1.102e-4,
              1.176e-4, 1.244e-4, 1.283e-4, 1.326e-4, 1.4e-4, 1.395e-4,
01805
              1.387e-4, 1.363e-4, 1.314e-4, 1.241e-4, 1.228e-4, 1.148e-4, 1.086e-4, 1.018e-4, 8.89e-5, 8.316e-5, 7.292e-5, 6.452e-5, 5.625e-5, 5.045e-5, 4.38e-5, 3.762e-5, 3.29e-5, 2.836e-5,
01806
01808
              2.485e-5, 2.168e-5, 1.895e-5, 1.659e-5, 1.453e-5, 1.282e-5,
01809
01810
              1.132e-5, 1.001e-5, 8.836e-6, 7.804e-6, 6.922e-6, 6.116e-6,
01811
              5.429e-6, 4.824e-6, 4.278e-6, 3.788e-6, 3.371e-6, 2.985e-6,
             2.649e-6, 2.357e-6, 2.09e-6, 1.858e-6, 1.647e-6, 1.462e-6, 1.299e-6, 1.155e-6, 1.028e-6, 9.142e-7, 8.132e-7, 7.246e-7,
01812
01813
              6.451e-7, 5.764e-7, 5.151e-7, 4.603e-7, 4.121e-7, 3.694e-7,
              3.318e-7, 2.985e-7, 2.69e-7, 2.428e-7, 2.197e-7, 1.992e-7,
01815
01816
              1.81e-7, 1.649e-7, 1.506e-7, 1.378e-7, 1.265e-7, 1.163e-7,
             1.073e-7, 9.918e-8, 9.191e-8, 8.538e-8, 7.949e-8, 7.419e-8, 6.94e-8, 6.508e-8, 6.114e-8, 5.761e-8, 5.437e-8, 5.146e-8, 4.89e-8, 4.636e-8, 4.406e-8, 4.201e-8, 4.015e-8, 3.84e-8, 3.661e-8, 3.51e-8, 3.377e-8, 3.242e-8, 3.13e-8, 3.015e-8,
01817
01818
01819
01820
              2.918e-8, 2.83e-8, 2.758e-8, 2.707e-8, 2.656e-8, 2.619e-8
01821
01822
              2.609e-8, 2.615e-8, 2.63e-8, 2.675e-8, 2.745e-8, 2.842e-8,
             2.966e-8, 3.125e-8, 3.318e-8, 3.565e-8, 3.85e-8, 4.191e-8, 4.59e-8, 5.059e-8, 5.607e-8, 6.239e-8, 6.958e-8, 7.796e-8, 8.773e-8, 9.88e-8, 1.114e-7, 1.258e-7, 1.422e-7, 1.61e-7, 1.822e-7, 2.06e-7, 2.337e-7, 2.645e-7, 2.996e-7, 3.393e-7,
01823
01824
01825
01827
              3.843e-7, 4.363e-7, 4.935e-7, 5.607e-7, 6.363e-7, 7.242e-7,
              8.23e-7, 9.411e-7, 1.071e-6, 1.232e-6, 1.402e-6, 1.6e-6, 1.82e-6,
01828
01829
              2.128e-6, 2.386e-6, 2.781e-6, 3.242e-6, 3.653e-6, 4.323e-6,
              4.747e-6, 5.321e-6, 5.919e-6, 6.681e-6, 7.101e-6, 7.983e-6,
01830
              8.342e-6, 8.741e-6, 9.431e-6, 9.952e-6, 1.026e-5, 1.055e-5,
01831
              1.095e-5, 1.095e-5, 1.087e-5, 1.056e-5, 1.026e-5, 9.715e-6,
01832
              9.252e-6, 8.452e-6, 7.958e-6, 7.268e-6, 6.295e-6, 6.003e-6, 5e-6,
01834
              4.591e-6, 3.983e-6, 3.479e-6, 3.058e-6, 2.667e-6, 2.293e-6,
01835
              1.995e-6, 1.747e-6, 1.517e-6, 1.335e-6, 1.165e-6, 1.028e-6,
             9.007e-7, 7.956e-7, 7.015e-7, 6.192e-7, 5.491e-7, 4.859e-7, 4.297e-7, 3.799e-7, 3.38e-7, 3.002e-7, 2.659e-7, 2.366e-7, 2.103e-7, 1.861e-7, 1.655e-7, 1.469e-7, 1.309e-7, 1.162e-7,
01836
01837
01838
              1.032e-7, 9.198e-8, 8.181e-8, 7.294e-8, 6.516e-8, 5.787e-8,
              5.163e-8, 4.612e-8, 4.119e-8, 3.695e-8, 3.308e-8, 2.976e-8,
01840
01841
              2.67e-8, 2.407e-8, 2.171e-8, 1.965e-8, 1.78e-8, 1.617e-8,
              1.47e-8, 1.341e-8, 1.227e-8, 1.125e-8, 1.033e-8, 9.524e-9, 8.797e-9, 8.162e-9, 7.565e-9, 7.04e-9, 6.56e-9, 6.129e-9, 5.733e-9, 5.376e-9, 5.043e-9, 4.75e-9, 4.466e-9, 4.211e-9,
01842
01843
01844
              3.977e-9, 3.759e-9, 3.558e-9, 3.373e-9, 3.201e-9, 3.043e-9,
              2.895e-9, 2.76e-9, 2.635e-9, 2.518e-9, 2.411e-9, 2.314e-9,
01846
              2.23e-9, 2.151e-9, 2.087e-9, 2.035e-9, 1.988e-9, 1.946e-9,
01847
              1.927e-9, 1.916e-9, 1.916e-9, 1.933e-9, 1.966e-9, 2.018e-9, 2.09e-9, 2.182e-9, 2.299e-9, 2.442e-9, 2.623e-9, 2.832e-9,
01848
01849
              3.079e-9, 3.368e-9, 3.714e-9, 4.104e-9, 4.567e-9, 5.091e-9, 5.701e-9, 6.398e-9, 7.194e-9, 8.127e-9, 9.141e-9, 1.035e-8,
01850
              1.177e-8, 1.338e-8, 1.508e-8, 1.711e-8, 1.955e-8, 2.216e-8,
01853
              2.534e-8, 2.871e-8, 3.291e-8, 3.711e-8, 4.285e-8, 4.868e-8,
             5.509e-8, 6.276e-8, 7.262e-8, 8.252e-8, 9.4e-8, 1.064e-7, 1.247e-7, 1.411e-7, 1.626e-7, 1.827e-7, 2.044e-7, 2.284e-7
01854
01855
              2.452e-7, 2.854e-7, 3.026e-7, 3.278e-7, 3.474e-7, 3.693e-7,
01856
01857
              3.93e-7, 4.104e-7, 4.22e-7, 4.439e-7, 4.545e-7, 4.778e-7,
              4.812e-7, 5.018e-7, 4.899e-7, 5.075e-7, 5.073e-7, 5.171e-
              5.131e-7, 5.25e-7, 5.617e-7, 5.846e-7, 6.239e-7, 6.696e-7, 7.398e-7, 8.073e-7, 9.15e-7, 1.009e-6, 1.116e-6, 1.264e-6,
01859
01860
              1.439e-6, 1.644e-6, 1.856e-6, 2.147e-6, 2.317e-6, 2.713e-6, 2.882e-6, 2.99e-6, 3.489e-6, 3.581e-6, 4.033e-6, 4.26e-6, 4.543e-6, 4.84e-6, 4.826e-6, 5.013e-6, 5.252e-6, 5.277e-6,
01861
01862
01863
              5.306e-6, 5.236e-6, 5.123e-6, 5.171e-6, 4.843e-6, 4.615e-6,
              4.385e-6, 3.97e-6, 3.693e-6, 3.231e-6, 2.915e-6, 2.495e-6,
01865
             2.144e-6, 1.91e-6, 1.639e-6, 1.417e-6, 1.226e-6, 1.065e-6, 9.29e-7, 8.142e-7, 7.161e-7, 6.318e-7, 5.581e-7, 4.943e-7, 4.376e-7, 3.884e-7, 3.449e-7, 3.06e-7, 2.712e-7, 2.412e-7, 2.139e-7, 1.903e-7, 1.689e-7, 1.499e-7, 1.331e-7, 1.183e-7, 1.05e-7, 9.362e-8, 8.306e-8, 7.403e-8, 6.578e-8, 5.853e-8,
01866
01867
01868
01869
              5.216e-8, 4.632e-8, 4.127e-8, 3.678e-8, 3.279e-8, 2.923e-8
01871
01872
              2.612e-8, 2.339e-8, 2.094e-8, 1.877e-8, 1.686e-8, 1.516e-8,
01873
              1.366e-8, 1.234e-8, 1.114e-8, 1.012e-8, 9.182e-9, 8.362e-9,
              7.634e-9, 6.981e-9, 6.406e-9, 5.888e-9, 5.428e-9, 5.021e-9, 4.65e-9, 4.326e-9, 4.033e-9, 3.77e-9, 3.536e-9, 3.327e-9,
01874
01875
```

```
3.141e-9, 2.974e-9, 2.825e-9, 2.697e-9, 2.584e-9, 2.488e-9,
            2.406e-9, 2.34e-9, 2.292e-9, 2.259e-9, 2.244e-9, 2.243e-9, 2.272e-9, 2.31e-9, 2.378e-9, 2.454e-9, 2.618e-9, 2.672e-9,
01877
01878
01879
            2.831e-9, 3.05e-9, 3.225e-9, 3.425e-9, 3.677e-9, 3.968e-9,
            4.221e-9, 4.639e-9, 4.96e-9, 5.359e-9, 5.649e-9, 6.23e-9, 6.716e-9, 7.218e-9, 7.746e-9, 7.988e-9, 8.627e-9, 8.999e-9,
01880
01881
            9.442e-9, 9.82e-9, 1.015e-8, 1.06e-8, 1.079e-8, 1.109e-8, 1.137e-8, 1.186e-8, 1.18e-8, 1.187e-8, 1.194e-8, 1.192e-8,
01883
01884
             1.224e-8, 1.245e-8, 1.246e-8, 1.318e-8, 1.377e-8, 1.471e-8,
01885
            1.582e-8, 1.713e-8, 1.853e-8, 2.063e-8, 2.27e-8, 2.567e-8,
             2.891e-8, 3.264e-8, 3.744e-8, 4.286e-8, 4.915e-8, 5.623e-8,
01886
             6.336e-8, 7.293e-8, 8.309e-8, 9.319e-8, 1.091e-7, 1.243e-7,
01887
            1.348e-7, 1.449e-7, 1.62e-7, 1.846e-7, 1.937e-7, 2.04e-7, 2.179e-7, 2.298e-7, 2.433e-7, 2.439e-7, 2.464e-7, 2.611e-7,
01888
01889
01890
            2.617e-7, 2.582e-7, 2.453e-7, 2.401e-7, 2.349e-7, 2.203e-7,
            2.066e-7, 1.939e-7, 1.78e-7, 1.558e-7, 1.391e-7, 1.203e-7, 1.048e-7, 9.464e-8, 8.306e-8, 7.239e-8, 6.317e-8, 5.52e-8, 4.847e-8, 4.282e-8, 3.796e-8, 3.377e-8, 2.996e-8, 2.678e-8,
01891
01892
01893
             2.4e-8, 2.134e-8, 1.904e-8, 1.705e-8, 1.523e-8, 1.35e-8,
             1.204e-8, 1.07e-8, 9.408e-9, 8.476e-9, 7.47e-9, 6.679e-9,
01895
            5.929e-9, 5.267e-9, 4.711e-9, 4.172e-9, 3.761e-9, 3.288e-9, 2.929e-9, 2.609e-9, 2.315e-9, 2.042e-9, 1.844e-9, 1.64e-9,
01896
01897
            1.47e-9, 1.31e-9, 1.176e-9, 1.049e-9, 9.377e-10, 8.462e-10, 7.616e-10, 6.854e-10, 6.191e-10, 5.596e-10, 5.078e-10, 4.611e-10, 4.197e-10, 3.83e-10, 3.505e-10, 3.215e-10, 2.956e-10, 2.726e-10,
01898
01899
01900
             2.521e-10, 2.338e-10, 2.173e-10, 2.026e-10, 1.895e-10, 1.777e-10,
01901
01902
             1.672e-10, 1.579e-10, 1.496e-10, 1.423e-10, 1.358e-10, 1.302e-10,
01903
            1.254e-10, 1.216e-10, 1.187e-10, 1.163e-10, 1.147e-10, 1.145e-10,
            1.15e-10, 1.17e-10, 1.192e-10, 1.25e-10, 1.298e-10, 1.345e-10, 1.405e-10, 1.538e-10, 1.648e-10, 1.721e-10, 1.872e-10, 1.968e-10, 2.089e-10, 2.172e-10, 2.317e-10, 2.389e-10, 2.503e-10, 2.585e-10,
01904
01905
01906
01907
             2.686e-10, 2.8e-10, 2.895e-10, 3.019e-10, 3.037e-10, 3.076e-10,
01908
             3.146e-10, 3.198e-10, 3.332e-10, 3.397e-10, 3.54e-10, 3.667e-10,
01909
            3.895e-10, 4.071e-10, 4.565e-10, 4.983e-10, 5.439e-10, 5.968e-10,
01910
             6.676e-10, 7.456e-10, 8.405e-10, 9.478e-10, 1.064e-9, 1.218e-9,
            1.386e-9, 1.581e-9, 1.787e-9, 2.032e-9, 2.347e-9, 2.677e-9,
01911
             3.008e-9, 3.544e-9, 4.056e-9, 4.687e-9, 5.331e-9, 6.227e-9,
01912
             6.854e-9, 8.139e-9, 8.945e-9, 9.865e-9, 1.125e-8, 1.178e-8,
01914
             1.364e-8, 1.436e-8, 1.54e-8, 1.672e-8, 1.793e-8, 1.906e-8,
01915
             2.036e-8, 2.144e-8, 2.292e-8, 2.371e-8, 2.493e-8, 2.606e-8,
01916
            2.706e-8, 2.866e-8, 3.036e-8, 3.136e-8, 3.405e-8, 3.665e-8,
             3.837e-8, 4.229e-8, 4.748e-8, 5.32e-8, 5.763e-8, 6.677e-8,
01917
            7.216e-8, 7.716e-8, 8.958e-8, 9.419e-8, 1.036e-7, 1.108e-7, 1.189e-7, 1.246e-7, 1.348e-7, 1.31e-7, 1.361e-7, 1.364e-7,
01918
01919
             1.363e-7, 1.343e-7, 1.293e-7, 1.254e-7, 1.235e-7, 1.158e-7,
01920
01921
             1.107e-7, 9.961e-8, 9.011e-8, 7.91e-8, 6.916e-8, 6.338e-8,
01922
            5.564e-8, 4.827e-8, 4.198e-8, 3.695e-8, 3.276e-8, 2.929e-8,
01923
            2.633e-8, 2.391e-8, 2.192e-8, 2.021e-8, 1.89e-8, 1.772e-8,
             1.667e-8, 1.603e-8, 1.547e-8, 1.537e-8, 1.492e-8, 1.515e-8,
01924
             1.479e-8, 1.45e-8, 1.513e-8, 1.495e-8, 1.529e-8, 1.565e-8,
01925
             1.564e-8, 1.553e-8, 1.569e-8, 1.584e-8, 1.57e-8, 1.538e-8,
             1.513e-8, 1.472e-8, 1.425e-8, 1.349e-8, 1.328e-8, 1.249e-8,
01927
            1.17e-8, 1.077e-8, 9.514e-9, 8.614e-9, 7.46e-9, 6.621e-9,
01928
            5.775e-9, 5.006e-9, 4.308e-9, 3.747e-9, 3.24e-9, 2.84e-9, 2.481e-9, 2.184e-9, 1.923e-9, 1.71e-9, 1.504e-9, 1.334e-9,
01929
01930
             1.187e-9, 1.053e-9, 9.367e-10, 8.306e-10, 7.419e-10, 6.63e-10,
01931
             5.918e-10, 5.277e-10, 4.717e-10, 4.222e-10, 3.783e-10, 3.39e-10,
             3.036e-10, 2.729e-10, 2.455e-10, 2.211e-10, 1.995e-10, 1.804e-10,
01933
01934
             1.635e-10, 1.485e-10, 1.355e-10, 1.24e-10, 1.139e-10, 1.051e-10,
            9.757e-11, 9.114e-11, 8.577e-11, 8.139e-11, 7.792e-11, 7.52e-11, 7.39e-11, 7.31le-11, 7.277e-11, 7.482e-11, 7.698e-11, 8.162e-11, 8.517e-11, 8.968e-11, 9.905e-11, 1.075e-10, 1.187e-10, 1.291e-10, 1.426e-10, 1.573e-10, 1.734e-10, 1.905e-10, 2.097e-10, 2.28e-10,
01935
01936
01937
             2.473e-10, 2.718e-10, 2.922e-10, 3.128e-10, 3.361e-10, 3.641e-10,
01939
01940
            3.91e-10, 4.196e-10, 4.501e-10, 4.932e-10, 5.258e-10, 5.755e-10,
01941
             6.253e-10, 6.664e-10, 7.344e-10, 7.985e-10, 8.877e-10, 1.005e-9,
01942
            1.118e-9, 1.251e-9, 1.428e-9, 1.61e-9, 1.888e-9, 2.077e-9, 2.331e-9, 2.751e-9, 3.061e-9, 3.522e-9, 3.805e-9, 4.181e-9,
01943
01944
             4.575e-9, 5.167e-9, 5.634e-9, 6.007e-9, 6.501e-9, 6.829e-9,
             7.211e-9, 7.262e-9, 7.696e-9, 7.832e-9, 7.799e-9, 7.651e-9,
01946
             7.304e-9, 7.15e-9, 6.977e-9, 6.603e-9, 6.209e-9, 5.69e-9,
01947
            5.432e-9, 4.764e-9, 4.189e-9, 3.64e-9, 3.203e-9, 2.848e-9,
            2.51e-9, 2.194e-9, 1.946e-9, 1.75e-9, 1.567e-9, 1.426e-9, 1.302e-9, 1.197e-9, 1.109e-9, 1.035e-9, 9.719e-10, 9.207e-10,
01948
01949
            7.741e-10, 7.762e-10, 7.537e-10, 7.424e-10, 7.474e-10, 7.294e-10,
01950
01951
01952
             7.216e-10, 7.233e-10, 7.075e-10, 6.892e-10, 6.618e-10, 6.314e-10,
01953
             6.208e-10, 5.689e-10, 5.55e-10, 4.984e-10, 4.6e-10, 4.078e-10,
            3.879e-10, 3.459e-10, 2.982e-10, 2.626e-10, 2.329e-10, 1.988e-10,
01954
             1.735e-10. 1.487e-10. 1.297e-10. 1.133e-10. 9.943e-11. 8.736e-11.
01955
            7.726e-11, 6.836e-11, 6.053e-11, 5.384e-11, 4.789e-11, 4.267e-11, 3.804e-11, 3.398e-11, 3.034e-11, 2.71e-11, 2.425e-11, 2.173e-11,
01956
             1.95e-11, 1.752e-11, 1.574e-11, 1.418e-11, 1.278e-11, 1.154e-11,
01958
01959
            1.044e-11, 9.463e-12, 8.602e-12, 7.841e-12, 7.171e-12, 6.584e-12,
01960
             6.073e-12, 5.631e-12, 5.254e-12, 4.937e-12, 4.679e-12, 4.476e-12,
            4.328e-12, 4.233e-12, 4.194e-12, 4.211e-12, 4.286e-12, 4.424e-12, 4.628e-12, 4.906e-12, 5.262e-12, 5.708e-12, 6.254e-12, 6.914e-12,
01961
01962
```

```
7.714e-12, 8.677e-12, 9.747e-12, 1.101e-11, 1.256e-11, 1.409e-11,
            1.597e-11, 1.807e-11, 2.034e-11, 2.316e-11, 2.622e-11, 2.962e-11, 3.369e-11, 3.819e-11, 4.329e-11, 4.932e-11, 5.589e-11, 6.364e-11,
01964
01965
01966
             7.284e-11, 8.236e-11, 9.447e-11, 1.078e-10, 1.229e-10, 1.417e-10,
01967
            1.614e-10, 1.843e-10, 2.107e-10, 2.406e-10, 2.728e-10, 3.195e-10,
             3.595e-10, 4.153e-10, 4.736e-10, 5.41e-10, 6.088e-10, 6.769e-10,
01968
             7.691e-10, 8.545e-10, 9.621e-10, 1.047e-9, 1.161e-9, 1.296e-9,
             1.424e-9, 1.576e-9, 1.739e-9, 1.893e-9, 2.08e-9, 2.336e-9,
01970
01971
            2.604e-9, 2.76e-9, 3.001e-9, 3.365e-9, 3.55e-9, 3.895e-9,
01972
             4.183e-9, 4.614e-9, 4.846e-9, 5.068e-9, 5.427e-9, 5.541e-9,
            5.864e-9, 5.997e-9, 5.997e-9, 6.061e-9, 5.944e-9, 5.855e-9,
01973
01974
            5.661e-9, 5.523e-9, 5.374e-9, 4.94e-9, 4.688e-9, 4.17e-9,
01975
             3.913e-9, 3.423e-9, 2.997e-9, 2.598e-9, 2.253e-9, 1.946e-9,
             1.71e-9, 1.507e-9, 1.336e-9, 1.19e-9, 1.068e-9, 9.623e-10,
01976
01977
            8.772e-10, 8.007e-10, 7.42e-10, 6.884e-10, 6.483e-10, 6.162e-10,
01978
             5.922e-10, 5.688e-10, 5.654e-10, 5.637e-10, 5.701e-10, 5.781e-10,
01979
            5.874e-10, 6.268e-10, 6.357e-10, 6.525e-10, 7.137e-10, 7.441e-10,
            8.024e-10, 8.485e-10, 9.143e-10, 9.536e-10, 9.717e-10, 1.018e-9,
01980
            1.042e-9, 1.054e-9, 1.092e-9, 1.079e-9, 1.064e-9, 1.043e-9,
             1.02e-9, 9.687e-10, 9.273e-10, 9.208e-10, 9.068e-10, 7.687e-10,
             7.385e-10, 6.595e-10, 5.87e-10, 5.144e-10, 4.417e-10, 3.804e-10,
01983
01984
            3.301e-10, 2.866e-10, 2.509e-10, 2.202e-10, 1.947e-10, 1.719e-10,
            1.525e-10, 1.361e-10, 1.21e-10, 1.084e-10, 9.8e-11, 8.801e-11,
01985
            7.954e-11, 7.124e-11, 6.335e-11, 5.76e-11, 5.132e-11, 4.601e-11, 4.096e-11, 3.657e-11, 3.25e-11, 2.909e-11, 2.587e-11, 2.297e-11,
01986
01987
            2.05e-11, 1.828e-11, 1.632e-11, 1.462e-11, 1.314e-11, 1.185e-11, 1.073e-11, 9.76e-12, 8.922e-12, 8.206e-12, 7.602e-12, 7.1e-12,
01989
01990
             6.694e-12, 6.378e-12, 6.149e-12, 6.004e-12, 5.941e-12, 5.962e-12,
01991
             6.069e-12, 6.265e-12, 6.551e-12, 6.935e-12, 7.457e-12, 8.074e-12,
01992
            8.811e-12, 9.852e-12, 1.086e-11, 1.207e-11, 1.361e-11, 1.553e-11,
            1.737e-11, 1.93e-11, 2.175e-11, 2.41e-11, 2.706e-11, 3.023e-11,
01993
01994
             3.313e-11, 3.657e-11, 4.118e-11, 4.569e-11, 5.025e-11, 5.66e-11,
             6.231e-11, 6.881e-11, 7.996e-11, 8.526e-11, 9.694e-11, 1.106e-10,
01995
01996
            1.222e-10, 1.355e-10, 1.525e-10, 1.775e-10, 1.924e-10, 2.181e-10,
01997
            2.379e-10, 2.662e-10, 2.907e-10, 3.154e-10, 3.366e-10, 3.579e-10,
            3.858e-10, 4.046e-10, 4.196e-10, 4.166e-10, 4.457e-10, 4.466e-10, 4.404e-10, 4.337e-10, 4.15e-10, 4.083e-10, 3.91e-10, 3.723e-10, 3.514e-10, 3.303e-10, 2.847e-10, 2.546e-10, 2.23e-10, 1.994e-10,
01998
01999
02001
             1.733e-10, 1.488e-10, 1.297e-10, 1.144e-10, 1.004e-10, 8.741e-11,
            7.928e-11, 7.034e-11, 6.323e-11, 5.754e-11, 5.25e-11, 4.85e-11,
02002
02003
            4.502e-11, 4.286e-11, 4.028e-11, 3.899e-11, 3.824e-11, 3.761e-11,
            3.804e-11, 3.839e-11, 3.845e-11, 4.244e-11, 4.382e-11, 4.582e-11,
02004
            4.847e-11, 5.209e-11, 5.384e-11, 5.887e-11, 6.371e-11, 6.737e-11, 7.168e-11, 7.415e-11, 7.827e-11, 8.037e-11, 8.12e-11, 8.071e-11,
02005
02006
            8.008e-11, 7.551e-11, 7.544e-11, 7.377e-11, 7.173e-11, 6.801e-11, 6.267e-11, 5.727e-11, 5.288e-11, 4.853e-11, 4.082e-11, 3.645e-11,
02007
02008
02009
            3.136e-11, 2.672e-11, 2.304e-11, 1.986e-11, 1.725e-11, 1.503e-11,
            1.315e-11, 1.153e-11, 1.014e-11, 8.942e-12, 7.901e-12, 6.993e-12, 6.199e-12, 5.502e-12, 4.89e-12, 4.351e-12, 3.878e-12, 3.461e-12, 3.094e-12, 2.771e-12, 2.488e-12, 2.241e-12, 2.025e-12, 1.838e-12,
02010
02011
02012
             1.677e-12, 1.541e-12, 1.427e-12, 1.335e-12, 1.262e-12, 1.209e-12,
            1.176e-12, 1.161e-12, 1.165e-12, 1.189e-12, 1.234e-12, 1.3e-12,
02014
02015
            1.389e-12, 1.503e-12, 1.644e-12, 1.814e-12, 2.017e-12, 2.255e-12,
            2.534e-12, 2.858e-12, 3.231e-12, 3.661e-12, 4.153e-12, 4.717e-12, 5.36e-12, 6.094e-12, 6.93e-12, 7.882e-12, 8.966e-12, 1.02e-11, 1.162e-11, 1.324e-11, 1.51e-11, 1.72e-11, 1.965e-11, 2.237e-11, 2.56e-11, 2.927e-11, 3.371e-11, 3.842e-11, 4.429e-11, 5.139e-11,
02016
02017
02018
             5.798e-11, 6.697e-11, 7.626e-11, 8.647e-11, 1.022e-10, 1.136e-10,
02020
             1.3e-10, 1.481e-10, 1.672e-10, 1.871e-10, 2.126e-10, 2.357e-10,
02021
02022
            2.583e-10, 2.997e-10, 3.289e-10, 3.702e-10, 4.012e-10, 4.319e-10,
02023
             4.527e-10, 5.001e-10, 5.448e-10, 5.611e-10, 5.76e-10, 5.965e-10,
            6.079e-10, 6.207e-10, 6.276e-10, 6.222e-10, 6.137e-10, 6e-10, 5.814e-10, 5.393e-10, 5.35e-10, 4.947e-10, 4.629e-10, 4.117e-10,
02024
            3.712e-10, 3.372e-10, 2.923e-10, 2.55e-10, 2.232e-10, 1.929e-10, 1.679e-10, 1.46e-10, 1.289e-10, 1.13e-10, 9.953e-11, 8.763e-11,
02026
02027
02028
            7.76e-11, 6.9e-11, 6.16e-11, 5.525e-11, 4.958e-11, 4.489e-11,
02029
            4.072e-11, 3.728e-11, 3.438e-11, 3.205e-11, 3.006e-11, 2.848e-11, 2.766e-11, 2.688e-11, 2.664e-11, 2.67e-11, 2.696e-11, 2.786e-11,
02030
            2.861e-11, 3.009e-11, 3.178e-11, 3.389e-11, 3.587e-11, 3.819e-11,
02031
             4.054e-11, 4.417e-11, 4.703e-11, 5.137e-11, 5.46e-11, 6.055e-11,
             6.333e-11, 6.773e-11, 7.219e-11, 7.717e-11, 8.131e-11, 8.491e-11,
02033
            8.574e-11, 9.01e-11, 9.017e-11, 8.999e-11, 8.959e-11, 8.838e-11,
02034
02035
            8.579e-11, 8.162e-11, 8.098e-11, 7.472e-11, 7.108e-11, 6.559e-11,
            5.994e-11, 5.172e-11, 4.424e-11, 3.951e-11, 3.34e-11, 2.902e-11, 2.541e-11, 2.215e-11, 1.945e-11, 1.716e-11, 1.503e-11, 1.339e-11, 1.185e-11, 1.05e-11, 9.336e-12, 8.307e-12, 7.312e-12, 6.55e-12, 5.836e-12, 5.178e-12, 4.6e-12, 4.086e-12, 3.639e-12, 3.247e-12,
02036
02037
02038
02039
02040
            2.904e-12, 2.604e-12, 2.341e-12, 2.112e-12, 1.914e-12, 1.744e-12,
02041
            1.598e-12, 1.476e-12, 1.374e-12, 1.293e-12, 1.23e-12, 1.185e-12,
            1.158e-12, 1.147e-12, 1.154e-12, 1.177e-12, 1.219e-12, 1.28e-12,
02042
            1.36e-12, 1.463e-12, 1.591e-12, 1.75e-12, 1.94e-12, 2.156e-12, 2.43e-12, 2.748e-12, 3.052e-12, 3.533e-12, 3.967e-12, 4.471e-12,
02043
             5.041e-12, 5.86e-12, 6.664e-12, 7.522e-12, 8.342e-12, 9.412e-12,
02045
02046
            1.072e-11, 1.213e-11, 1.343e-11, 1.496e-11, 1.664e-11, 1.822e-11,
02047
            2.029e-11, 2.233e-11, 2.457e-11, 2.709e-11, 2.928e-11, 3.115e-11,
            3.356e-11, 3.592e-11, 3.818e-11, 3.936e-11, 4.061e-11, 4.149e-11, 4.299e-11, 4.223e-11, 4.251e-11, 4.287e-11, 4.177e-11, 4.094e-11,
02048
02049
```

```
3.942e-11, 3.772e-11, 3.614e-11, 3.394e-11, 3.222e-11, 2.791e-11,
             2.665e-11, 2.309e-11, 2.032e-11, 1.74e-11, 1.535e-11, 1.323e-11, 1.151e-11, 9.803e-12, 8.65e-12, 7.54e-12, 6.619e-12, 5.832e-12,
02051
02052
02053
             5.113e-12, 4.503e-12, 3.975e-12, 3.52e-12, 3.112e-12, 2.797e-12,
             2.5e-12, 2.24e-12, 2.013e-12, 1.819e-12, 1.653e-12, 1.513e-12, 1.395e-12, 1.299e-12, 1.225e-12, 1.168e-12, 1.124e-12, 1.148e-12,
02054
02055
             1.107e-12, 1.128e-12, 1.169e-12, 1.233e-12, 1.307e-12, 1.359e-12,
             1.543e-12, 1.686e-12, 1.794e-12, 2.028e-12, 2.21e-12, 2.441e-12,
02057
02058
             2.653e-12, 2.828e-12, 3.093e-12, 3.28e-12, 3.551e-12, 3.677e-12,
02059
             3.803e-12, 3.844e-12, 4.068e-12, 4.093e-12, 4.002e-12, 3.904e-12,
02060
             3.624e-12, 3.633e-12, 3.622e-12, 3.443e-12, 3.184e-12, 2.934e-12,
             2.476e-12, 2.212e-12, 1.867e-12, 1.594e-12, 1.37e-12, 1.192e-12, 1.045e-12, 9.211e-13, 8.17e-13, 7.29e-13, 6.55e-13, 5.929e-13,
02061
02062
             5.415e-13, 4.995e-13, 4.661e-13, 4.406e-13, 4.225e-13, 4.116e-13,
02063
02064
             4.075e-13, 4.102e-13, 4.198e-13, 4.365e-13, 4.606e-13, 4.925e-13,
02065
             5.326e-13, 5.818e-13, 6.407e-13, 7.104e-13, 7.92e-13, 8.868e-13,
             9.964e-13, 1.123e-12, 1.268e-12, 1.434e-12, 1.626e-12, 1.848e-12,
02066
             2.107e-12, 2.422e-12, 2.772e-12, 3.145e-12, 3.704e-12, 4.27e-12,
02067
             4.721e-12, 5.361e-12, 6.083e-12, 7.095e-12, 7.968e-12, 9.228e-12,
             1.048e-11, 1.187e-11, 1.336e-11, 1.577e-11, 1.772e-11, 2.017e-11,
02069
             2.25e-11, 2.63e-11, 2.911e-11, 3.356e-11, 3.82e-11, 4.173e-11,
02070
02071
             4.811e-11, 5.254e-11, 5.839e-11, 6.187e-11, 6.805e-11, 7.118e-11,
             7.369e-11, 7.664e-11, 7.794e-11, 7.947e-11, 8.036e-11, 7.954e-11,
02072
02073
             7.849e-11, 7.518e-11, 7.462e-11, 6.926e-11, 6.531e-11, 6.197e-11, 5.421e-11, 4.777e-11, 4.111e-11, 3.679e-11, 3.166e-11, 2.786e-11,
02074
02075
             2.436e-11, 2.144e-11, 1.859e-11, 1.628e-11, 1.414e-11, 1.237e-11,
02076
             1.093e-11, 9.558e-12
02077
02078
          static double h2o260[2001] = { .2752, .2732, .2749, .2676, .2667, .2545, .2497, .2327, .2218, .2036, .1825, .1694, .1497, .1353, .121, .1014, .09405, .07848, .07195, .06246, .05306, .04853, .04138, .03735, .03171, .02785, .02431, .02111, .01845, .0164, .01405,
02079
02080
02082
02083
             .01255, .01098, .009797, .008646, .007779, .006898, .006099,
             .005453, .004909, .004413, .003959, .003581, .003199, .002871, .002583, .00233, .002086, .001874, .001684, .001512, .001361, .001225, .0011, 9.89e-4, 8.916e-4, 8.039e-4, 7.256e-4, 6.545e-4,
02084
02085
02086
             5.918e-4, 5.359e-4, 4.867e-4, 4.426e-4, 4.033e-4, 3.682e-4,
02088
             3.366e-4, 3.085e-4, 2.833e-4, 2.605e-4, 2.403e-4, 2.221e-4,
             2.055e-4, 1.908e-4, 1.774e-4, 1.653e-4, 1.544e-4, 1.443e-4,
02089
             1.351e-4, 1.267e-4, 1.19e-4, 1.119e-4, 1.053e-4, 9.922e-5, 9.355e-5, 8.831e-5, 8.339e-5, 7.878e-5, 7.449e-5, 7.043e-5, 6.664e-5, 6.307e-5, 5.969e-5, 5.654e-5, 5.357e-5, 5.075e-5, 4.81e-5, 4.56e-5, 4.322e-5, 4.102e-5, 3.892e-5, 3.696e-5,
02090
02091
02092
02093
             3.511e-5, 3.339e-5, 3.177e-5, 3.026e-5, 2.886e-5, 2.756e-5,
02094
02095
             2.636e-5, 2.527e-5, 2.427e-5, 2.337e-5, 2.257e-5, 2.185e-5
02096
             2.127e-5, 2.08e-5, 2.041e-5, 2.013e-5, 2e-5, 1.997e-5, 2.009e-5,
02097
             2.031e-5, 2.068e-5, 2.124e-5, 2.189e-5, 2.267e-5, 2.364e-5,
             2.463e-5, 2.618e-5, 2.774e-5, 2.937e-5, 3.144e-5, 3.359e-5,
02098
02099
             3.695e-5, 4.002e-5, 4.374e-5, 4.947e-5, 5.431e-5, 6.281e-5,
             7.169e-5, 8.157e-5, 9.728e-5, 1.079e-4, 1.337e-4, 1.442e-4,
             1.683e-4, 1.879e-4, 2.223e-4, 2.425e-4, 2.838e-4, 3.143e-4
02101
02102
             3.527e-4, 4.012e-4, 4.237e-4, 4.747e-4, 5.057e-4, 5.409e-4,
02103
             5.734e-4, 5.944e-4, 6.077e-4, 6.175e-4, 6.238e-4, 6.226e-4,
             6.248e-4, 6.192e-4, 6.098e-4, 5.818e-4, 5.709e-4, 5.465e-4, 5.043e-4, 4.699e-4, 4.294e-4, 3.984e-4, 3.672e-4, 3.152e-4,
02104
02105
             2.883e-4, 2.503e-4, 2.211e-4, 1.92e-4, 1.714e-4, 1.485e-4,
             1.358e-4, 1.156e-4, 1.021e-4, 8.887e-5, 7.842e-5, 7.12e-5,
02107
             6.186e-5, 5.73e-5, 4.792e-5, 4.364e-5, 3.72e-5, 3.28e-5, 2.946e-5, 2.591e-5, 2.261e-5, 2.048e-5, 1.813e-5, 1.63e-5,
02108
02109
02110
             1.447e-5, 1.282e-5, 1.167e-5, 1.041e-5, 9.449e-6, 8.51e-6,
             7.596e-6, 6.961e-6, 6.272e-6, 5.728e-6, 5.198e-6, 4.667e-6, 4.288e-6, 3.897e-6, 3.551e-6, 3.235e-6, 2.952e-6, 2.688e-6,
02111
             2.449e-6, 2.241e-6, 2.05e-6, 1.879e-6, 1.722e-6, 1.582e-6,
02113
02114
             1.456e-6, 1.339e-6, 1.236e-6, 1.144e-6, 1.06e-6, 9.83e-7,
02115
             9.149e-7, 8.535e-7, 7.973e-7, 7.466e-7, 6.999e-7, 6.574e-7,
             6.18e-7, 5.821e-7, 5.487e-7, 5.18e-7, 4.896e-7, 4.631e-7, 4.386e-7, 4.16e-7, 3.945e-7, 3.748e-7, 3.562e-7, 3.385e-7, 3.222e-7, 3.068e-7, 2.922e-7, 2.788e-7, 2.659e-7, 2.539e-7,
02116
02117
02118
             2.425e-7, 2.318e-7, 2.219e-7, 2.127e-7, 2.039e-7, 1.958e-7,
             1.885e-7, 1.818e-7, 1.758e-7, 1.711e-7, 1.662e-7, 1.63e-7,
02120
02121
             1.605e-7, 1.58e-7, 1.559e-7, 1.545e-7, 1.532e-7, 1.522e-7,
             1.51e-7, 1.495e-7, 1.465e-7, 1.483e-7, 1.469e-7, 1.448e-7, 1.444e-7, 1.436e-7, 1.426e-7, 1.431e-7, 1.425e-7, 1.445e-7, 1.477e-7, 1.515e-7, 1.567e-7, 1.634e-7, 1.712e-7, 1.802e-7,
02122
02123
02124
             1.914e-7, 2.024e-7, 2.159e-7, 2.295e-7, 2.461e-7, 2.621e-7,
02125
             2.868e-7, 3.102e-7, 3.394e-7, 3.784e-7, 4.223e-7, 4.864e-7,
02126
02127
             5.501e-7, 6.039e-7, 7.193e-7, 7.728e-7, 9.514e-7, 1.073e-6,
             1.18e-6, 1.333e-6, 1.472e-6, 1.566e-6, 1.677e-6, 1.784e-6, 1.904e-6, 1.953e-6, 2.02e-6, 2.074e-6, 2.128e-6, 2.162e-6,
02128
02129
             2.219e-6, 2.221e-6, 2.249e-6, 2.239e-6, 2.235e-6, 2.185e-6, 2.141e-6, 2.124e-6, 2.09e-6, 2.068e-6, 2.1e-6, 2.104e-6,
02130
             2.142e-6, 2.181e-6, 2.257e-6, 2.362e-6, 2.5e-6, 2.664e-6,
02132
02133
             2.884e-6, 3.189e-6, 3.48e-6, 3.847e-6, 4.313e-6, 4.79e-6,
             5.25e-6, 5.989e-6, 6.692e-6, 7.668e-6, 8.52e-6, 9.606e-6, 1.073e-5, 1.225e-5, 1.377e-5, 1.582e-5, 1.761e-5, 2.029e-5, 2.284e-5, 2.602e-5, 2.94e-5, 3.483e-5, 3.928e-5, 4.618e-5,
02134
02135
02136
```

```
5.24e-5, 6.132e-5, 7.183e-5, 8.521e-5, 9.111e-5, 1.07e-4,
              1.184e-4, 1.264e-4, 1.475e-4, 1.612e-4, 1.704e-4, 1.818e-4, 1.924e-4, 1.994e-4, 2.061e-4, 2.18e-4, 2.187e-4, 2.2e-4,
02138
02139
02140
              2.196e-4, 2.131e-4, 2.015e-4, 1.988e-4, 1.847e-4, 1.729e-4,
              1.597e-4, 1.373e-4, 1.262e-4, 1.087e-4, 9.439e-5, 8.061e-5, 7.093e-5, 6.049e-5, 5.12e-5, 4.435e-5, 3.817e-5, 3.34e-5,
02141
02142
              2.927e-5, 2.573e-5, 2.291e-5, 2.04e-5, 1.827e-5, 1.636e-5,
              1.463e-5, 1.309e-5, 1.17e-5, 1.047e-5, 9.315e-6, 8.328e-6,
02144
02145
              7.458e-6, 6.665e-6, 5.94e-6, 5.316e-6, 4.752e-6, 4.252e-6,
             7.458e-6, 6.665e-6, 5.94e-6, 5.316e-6, 4.752e-6, 4.252e-6, 3.825e-6, 3.421e-6, 3.064e-6, 2.746e-6, 2.465e-6, 2.216e-6, 1.99e-6, 1.79e-6, 1.609e-6, 1.449e-6, 1.306e-6, 1.177e-6, 1.063e-6, 9.607e-7, 8.672e-7, 7.855e-7, 7.118e-7, 6.46e-7, 5.871e-7, 5.34e-7, 4.868e-7, 4.447e-7, 4.068e-7, 3.729e-7, 3.423e-7, 3.151e-7, 2.905e-7, 2.686e-7, 2.484e-7, 2.306e-7, 2.142e-7, 1.995e-7, 1.86e-7, 1.738e-7, 1.626e-7, 1.522e-7, 1.427e-7, 1.338e-7, 1.258e-7, 1.118e-7, 1.056e-7, 9.972e-8, 9.46e-8, 9.007e-8, 8.592e-8, 8.195e-8, 7.816e-8, 7.483e-8, 7.193e-8, 6.892e-8, 6.642e-8, 6.386e-8, 6.154e-8,
02146
02147
02148
02149
02150
02151
02152
02153
              7.483e-8, 7.193e-8, 6.892e-8, 6.642e-8, 6.386e-8, 6.154e-8,
02154
              5.949e-8, 5.764e-8, 5.622e-8, 5.479e-8, 5.364e-8, 5.301e-8,
              5.267e-8, 5.263e-8, 5.313e-8, 5.41e-8, 5.55e-8, 5.745e-8,
02156
              6.003e-8, 6.311e-8, 6.713e-8, 7.173e-8, 7.724e-8, 8.368e-8, 9.121e-8, 9.986e-8, 1.097e-7, 1.209e-7, 1.338e-7, 1.486e-7,
02157
02158
              1.651e-7, 1.837e-7, 2.048e-7, 2.289e-7, 2.557e-7, 2.857e-7,
02159
              3.195e-7, 3.587e-7, 4.015e-7, 4.497e-7, 5.049e-7, 5.665e-7, 6.366e-7, 7.121e-7, 7.996e-7, 8.946e-7, 1.002e-6, 1.117e-6,
02160
02161
02162
              1.262e-6, 1.416e-6, 1.611e-6, 1.807e-6, 2.056e-6, 2.351e-6,
              2.769e-6, 3.138e-6, 3.699e-6, 4.386e-6, 5.041e-6, 6.074e-6,
02163
             6.812e-6, 7.79e-6, 8.855e-6, 1.014e-5, 1.095e-5, 1.245e-5, 1.316e-5, 1.39e-5, 1.504e-5, 1.583e-5, 1.617e-5, 1.652e-5,
02164
02165
02166
              1.713e-5, 1.724e-5, 1.715e-5, 1.668e-5, 1.629e-5, 1.552e-5,
              1.478e-5, 1.34e-5, 1.245e-5, 1.121e-5, 9.575e-6, 8.956e-6,
02167
02168
              7.345e-6, 6.597e-6, 5.612e-6, 4.818e-6, 4.165e-6, 3.579e-6,
              3.041e-6, 2.623e-6, 2.29e-6, 1.984e-6, 1.748e-6, 1.534e-6,
02169
02170
              1.369e-6, 1.219e-6, 1.092e-6, 9.8e-7, 8.762e-7, 7.896e-7,
             7.104e-7, 6.364e-7, 5.691e-7, 5.107e-7, 4.575e-7, 4.09e-7, 3.667e-7, 3.287e-7, 2.931e-7, 2.633e-7, 2.356e-7, 2.111e-7,
02171
02172
              1.895e-7, 1.697e-7, 1.525e-7, 1.369e-7, 1.233e-7, 1.114e-7,
02173
              9.988e-8, 9.004e-8, 8.149e-8, 7.352e-8, 6.662e-8, 6.03e-8,
02175
              5.479e-8, 4.974e-8, 4.532e-8, 4.129e-8, 3.781e-8, 3.462e-8,
              3.176e-8, 2.919e-8, 2.687e-8, 2.481e-8, 2.292e-8, 2.119e-8,
02176
02177
              1.967e-8, 1.828e-8, 1.706e-8, 1.589e-8, 1.487e-8, 1.393e-8,
              1.307e-8, 1.228e-8, 1.156e-8, 1.089e-8, 1.028e-8, 9.696e-9, 9.159e-9, 8.658e-9, 8.187e-9, 7.746e-9, 7.34e-9, 6.953e-9, 6.594e-9, 6.259e-9, 5.948e-9, 5.66e-9, 5.386e-9, 5.135e-9,
02178
02179
02180
              4.903e-9, 4.703e-9, 4.515e-9, 4.362e-9, 4.233e-9, 4.117e-9,
02181
02182
              4.017e-9, 3.962e-9, 3.924e-9, 3.905e-9, 3.922e-9, 3.967e-9,
02183
              4.046e-9, 4.165e-9, 4.32e-9, 4.522e-9, 4.769e-9, 5.083e-9,
02184
              5.443e-9, 5.872e-9, 6.366e-9, 6.949e-9, 7.601e-9, 8.371e-9,
              9.22e-9, 1.02e-8, 1.129e-8, 1.251e-8, 1.393e-8, 1.542e-8, 1.72e-8, 1.926e-8, 2.152e-8, 2.392e-8, 2.678e-8, 3.028e-8,
02185
02186
              3.39e-8, 3.836e-8, 4.309e-8, 4.9e-8, 5.481e-8, 6.252e-8,
              7.039e-8, 7.883e-8, 8.849e-8, 1.012e-7, 1.142e-7, 1.3e-7, 1.475e-7, 1.732e-7, 1.978e-7, 2.304e-7, 2.631e-7, 2.988e-7,
02188
02189
             1.43e-7, 1.33e-7, 2.33de-7, 2.33de-7, 2.33de-7, 2.33de-7, 3.392e-7, 3.69e-7, 4.355e-7, 4.672e-7, 5.11e-7, 5.461e-7, 5.828e-7, 6.233e-7, 6.509e-7, 6.672e-7, 6.969e-7, 7.104e-7, 7.439e-7, 7.463e-7, 7.708e-7, 7.466e-7, 7.668e-7, 7.549e-7, 7.586e-7, 7.384e-7, 7.439e-7, 7.785e-7, 7.915e-7, 8.31e-7,
02190
02191
02192
              8.745e-7, 9.558e-7, 1.038e-6, 1.173e-6, 1.304e-6, 1.452e-6,
02194
02195
              1.671e-6, 1.931e-6, 2.239e-6, 2.578e-6, 3.032e-6, 3.334e-6,
02196
              3.98e-6, 4.3e-6, 4.518e-6, 5.321e-6, 5.508e-6, 6.211e-6, 6.59e-6,
              7.046e-6, 7.555e-6, 7.558e-6, 7.875e-6, 8.319e-6, 8.433e-6, 8.59e-6, 8.503e-6, 8.304e-6, 8.336e-6, 7.739e-6, 7.301e-6,
02197
02198
02199
              6.827e-6, 6.078e-6, 5.551e-6, 4.762e-6, 4.224e-6, 3.538e-6,
              2.984e-6, 2.619e-6, 2.227e-6, 1.923e-6, 1.669e-6, 1.462e-6,
02200
02201
              1.294e-6, 1.155e-6, 1.033e-6, 9.231e-7, 8.238e-7, 7.36e-7,
02202
              6.564e-7, 5.869e-7, 5.236e-7, 4.673e-7, 4.174e-7, 3.736e-7,
              3.33e-7, 2.976e-7, 2.657e-7, 2.367e-7, 2.106e-7, 1.877e-7, 1.671e-7, 1.494e-7, 1.332e-7, 1.192e-7, 1.065e-7, 9.558e-8,
02203
02204
              8.586e-8, 7.717e-8, 6.958e-8, 6.278e-8, 5.666e-8, 5.121e-8,
02205
              4.647e-8, 4.213e-8, 3.815e-8, 3.459e-8, 3.146e-8, 2.862e-8,
              2.604e-8, 2.375e-8, 2.162e-8, 1.981e-8, 1.817e-8, 1.67e-8,
02207
02208
              1.537e-8, 1.417e-8, 1.31e-8, 1.215e-8, 1.128e-8, 1.05e-8,
              9.793e-9, 9.158e-9, 8.586e-9, 8.068e-9, 7.595e-9, 7.166e-9, 6.778e-9, 6.427e-9, 6.108e-9, 5.826e-9, 5.571e-9, 5.347e-9,
02209
02210
              5.144e-9, 4.968e-9, 4.822e-9, 4.692e-9, 4.589e-9, 4.506e-9,
02211
              4.467e-9, 4.44e-9, 4.466e-9, 4.515e-9, 4.718e-9, 4.729e-9,
02212
              4.937e-9, 5.249e-9, 5.466e-9, 5.713e-9, 6.03e-9, 6.436e-9,
02213
02214
              6.741e-9, 7.33e-9, 7.787e-9, 8.414e-9, 8.908e-9, 9.868e-9,
02215
              1.069e-8, 1.158e-8, 1.253e-8, 1.3e-8, 1.409e-8, 1.47e-8,
              1.548-8, 1.612e-8, 1.666e-8, 1.736e-8, 1.763e-8, 1.812e-8, 1.852e-8, 1.923e-8, 1.897e-8, 1.893e-8, 1.888e-8, 1.868e-8,
02216
02217
              1.895e-8, 1.899e-8, 1.876e-8, 1.96e-8, 2.02e-8, 2.121e-8,
              2.239e-8, 2.379e-8, 2.526e-8, 2.766e-8, 2.994e-8, 3.332e-8,
02219
02220
              3.703e-8, 4.158e-8, 4.774e-8, 5.499e-8, 6.355e-8, 7.349e-8,
02221
              8.414e-8, 9.846e-8, 1.143e-7, 1.307e-7, 1.562e-7, 1.817e-7,
              2.011e-7, 2.192e-7, 2.485e-7, 2.867e-7, 3.035e-7, 3.223e-7, 3.443e-7, 3.617e-7, 3.793e-7, 3.793e-7, 3.839e-7, 4.081e-7,
02222
02223
```

```
02224
            4.117e-7, 4.085e-7, 3.92e-7, 3.851e-7, 3.754e-7, 3.49e-7,
            3.229e-7, 2.978e-7, 2.691e-7, 2.312e-7, 2.029e-7, 1.721e-7, 1.472e-7, 1.308e-7, 1.132e-7, 9.736e-8, 8.458e-8, 7.402e-8,
02225
02226
02227
            6.534e-8, 5.811e-8, 5.235e-8, 4.762e-8, 4.293e-8, 3.896e-8,
            3.526e-8, 3.165e-8, 2.833e-8, 2.551e-8, 2.288e-8, 2.036e-8, 1.82e-8, 1.626e-8, 1.438e-8, 1.299e-8, 1.149e-8, 1.03e-8,
02228
02229
            9.148e-9, 8.122e-9, 7.264e-9, 6.425e-9, 5.777e-9, 5.06e-9,
            4.502e-9, 4.013e-9, 3.567e-9, 3.145e-9, 2.864e-9, 2.553e-9
02231
02232
            2.311e-9, 2.087e-9, 1.886e-9, 1.716e-9, 1.556e-9, 1.432e-9,
            1.311e-9, 1.202e-9, 1.104e-9, 1.013e-9, 9.293e-10, 8.493e-10,
02233
            7.79e-10, 7.185e-10, 6.642e-10, 6.141e-10, 5.684e-10, 5.346e-10,
02234
            5.032e-10, 4.725e-10, 4.439e-10, 4.176e-10, 3.93e-10, 3.714e-10, 3.515e-10, 3.332e-10, 3.167e-10, 3.02e-10, 2.887e-10, 2.769e-10,
02235
02236
            2.665e-10, 2.578e-10, 2.503e-10, 2.436e-10, 2.377e-10, 2.342e-10,
02237
02238
            2.305e-10, 2.296e-10, 2.278e-10, 2.321e-10, 2.355e-10, 2.402e-10,
02239
            2.478e-10, 2.67e-10, 2.848e-10, 2.982e-10, 3.263e-10, 3.438e-10,
            3.649e-10, 3.829e-10, 4.115e-10, 4.264e-10, 4.473e-10, 4.63e-10,
02240
            4.808e-10, 4.995e-10, 5.142e-10, 5.313e-10, 5.318e-10, 5.358e-10,
02241
            5.452e-10, 5.507e-10, 5.698e-10, 5.782e-10, 5.983e-10, 6.164e-10,
            6.532e-10, 6.811e-10, 7.624e-10, 8.302e-10, 9.067e-10,
02243
            1.104e-9, 1.221e-9, 1.361e-9, 1.516e-9, 1.675e-9, 1.883e-9,
02244
02245
            2.101e-9, 2.349e-9, 2.614e-9, 2.92e-9, 3.305e-9, 3.724e-9,
            4.142e-9, 4.887e-9, 5.614e-9, 6.506e-9, 7.463e-9, 8.817e-9,
02246
            9.849e-9, 1.187e-8, 1.321e-8, 1.474e-8, 1.698e-8, 1.794e-8, 2.09e-8, 2.211e-8, 2.362e-8, 2.556e-8, 2.729e-8, 2.88e-8,
02247
02248
            3.046e-8, 3.167e-8, 3.367e-8, 3.457e-8, 3.59e-8, 3.711e-8,
02249
02250
            3.826e-8, 4.001e-8, 4.211e-8, 4.315e-8, 4.661e-8, 5.01e-8,
            5.249e-8, 5.84e-8, 6.628e-8, 7.512e-8, 8.253e-8, 9.722e-8, 1.067e-7, 1.153e-7, 1.347e-7, 1.428e-7, 1.577e-7, 1.694e-7, 1.833e-7, 1.938e-7, 2.108e-7, 2.059e-7, 2.157e-7, 2.185e-7,
02251
02252
02253
02254
            2.208e-7, 2.182e-7, 2.093e-7, 2.014e-7, 1.962e-7, 1.819e-7,
            1.713e-7, 1.51e-7, 1.34e-7, 1.154e-7, 9.89e-8, 8.88e-8, 7.673e-8,
            6.599e-8, 5.73e-8, 5.081e-8, 4.567e-8, 4.147e-8, 3.773e-8,
02256
02257
            3.46e-8, 3.194e-8, 2.953e-8, 2.759e-8, 2.594e-8, 2.442e-8
            2.355e-8, 2.283e-8, 2.279e-8, 2.231e-8, 2.279e-8, 2.239e-8, 2.21e-8, 2.309e-8, 2.293e-8, 2.352e-8, 2.415e-8, 2.43e-8, 2.426e-8, 2.465e-8, 2.5e-8, 2.496e-8, 2.465e-8, 2.445e-8,
02258
02259
02260
            2.383e-8, 2.299e-8, 2.165e-8, 2.113e-8, 1.968e-8, 1.819e-8,
02262
            1.644e-8, 1.427e-8, 1.27e-8, 1.082e-8, 9.428e-9, 8.091e-9,
            6.958e-9, 5.988e-9, 5.246e-9, 4.601e-9, 4.098e-9, 3.664e-9,
02263
02264
            3.287e-9, 2.942e-9, 2.656e-9, 2.364e-9, 2.118e-9, 1.903e-9,
            1.703e-9, 1.525e-9, 1.365e-9, 1.229e-9, 1.107e-9, 9.96e-10, 8.945e-10, 8.08e-10, 7.308e-10, 6.616e-10, 5.994e-10, 5.422e-10, 4.929e-10, 4.478e-10, 4.07e-10, 3.707e-10, 3.379e-10, 3.087e-10,
02265
02266
02267
            2.823e-10, 2.592e-10, 2.385e-10, 2.201e-10, 2.038e-10, 1.897e-10
02268
02269
            1.774e-10, 1.667e-10, 1.577e-10, 1.502e-10, 1.437e-10, 1.394e-10,
02270
            1.358e-10, 1.324e-10, 1.329e-10, 1.324e-10, 1.36e-10, 1.39e-10,
02271
            1.424e-10, 1.544e-10, 1.651e-10, 1.817e-10, 1.984e-10, 2.195e-10,
            2.438e-10, 2.7e-10, 2.991e-10, 3.322e-10, 3.632e-10, 3.957e-10, 4.36e-10, 4.701e-10, 5.03e-10, 5.381e-10, 5.793e-10, 6.19e-10,
02272
02273
            6.596e-10, 7.004e-10, 7.561e-10, 7.934e-10, 8.552e-10, 9.142e-10,
            9.57e-10, 1.027e-9, 1.097e-9, 1.193e-9, 1.334e-9, 1.47e-9,
02275
02276
            1.636e-9, 1.871e-9, 2.122e-9, 2.519e-9, 2.806e-9, 3.203e-9,
02277
            3.846e-9, 4.362e-9, 5.114e-9, 5.643e-9, 6.305e-9, 6.981e-9,
            7.983e-9, 8.783e-9, 9.419e-9, 1.017e-8, 1.063e-8, 1.121e-8, 1.13e-8, 1.201e-8, 1.225e-8, 1.232e-8, 1.223e-8, 1.177e-8,
02278
02279
            1.151e-8, 1.116e-8, 1.047e-8, 9.698e-9, 8.734e-9, 8.202e-9,
            7.041e-9, 6.074e-9, 5.172e-9, 4.468e-9, 3.913e-9, 3.414e-9,
02281
            2.975e-9, 2.65e-9, 2.406e-9, 2.173e-9, 2.009e-9, 1.861e-9, 1.727e-9, 1.612e-9, 1.514e-9, 1.43e-9, 1.362e-9, 1.333e-9,
02282
02283
            1.288e-9, 1.249e-9, 1.238e-9, 1.217e-9, 1.202e-9, 1.209e-9, 1.177e-9, 1.157e-9, 1.165e-9, 1.142e-9, 1.131e-9,
02284
02285
            1.138e-9, 1.117e-9, 1.1e-9, 1.069e-9, 1.023e-9, 1.005e-9,
            9.159e-10, 8.863e-10, 7.865e-10, 7.153e-10, 6.247e-10, 5.846e-10, 5.133e-10, 4.36e-10, 3.789e-10, 3.335e-10, 2.833e-10, 2.483e-10,
02287
02288
02289
            2.155e-10, 1.918e-10, 1.709e-10, 1.529e-10, 1.374e-10, 1.235e-10,
02290
            1.108 e^{-10}, \ 9.933 e^{-11}, \ 8.932 e^{-11}, \ 8.022 e^{-11}, \ 7.224 e^{-11}, \ 6.52 e^{-11},
02291
            5.896e-11, 5.328e-11, 4.813e-11, 4.365e-11, 3.961e-11, 3.594e-11,
02292
            3.266e-11, 2.967e-11, 2.701e-11, 2.464e-11, 2.248e-11, 2.054e-11,
02293
            1.878e-11, 1.721e-11, 1.579e-11, 1.453e-11, 1.341e-11, 1.241e-11,
02294
            1.154e-11, 1.078e-11, 1.014e-11, 9.601e-12, 9.167e-12, 8.838e-12,
02295
            8.614e-12, 8.493e-12, 8.481e-12, 8.581e-12, 8.795e-12, 9.131e-12,
02296
            9.601e-12, 1.021e-11, 1.097e-11, 1.191e-11, 1.303e-11, 1.439e-11,
02297
            1.601e-11, 1.778e-11, 1.984e-11, 2.234e-11, 2.474e-11, 2.766e-11,
            3.085e-11, 3.415e-11, 3.821e-11, 4.261e-11, 4.748e-11, 5.323e-11,
02298
            5.935e-11, 6.619e-11, 7.418e-11, 8.294e-11, 9.26e-11, 1.039e-10,
02299
02300
            1.156e-10, 1.297e-10, 1.46e-10, 1.641e-10, 1.858e-10, 2.1e-10,
02301
            2.383e-10, 2.724e-10, 3.116e-10, 3.538e-10, 4.173e-10, 4.727e-10, 5.503e-10, 6.337e-10, 7.32e-10, 8.298e-10, 9.328e-10, 1.059e-9,
02302
            1.176e-9, 1.328e-9, 1.445e-9, 1.593e-9, 1.77e-9, 1.954e-9, 2.175e-9, 2.405e-9, 2.622e-9, 2.906e-9, 3.294e-9, 3.713e-9,
02303
02304
            3.98e-9, 4.384e-9, 4.987e-9, 5.311e-9, 5.874e-9, 6.337e-9,
            7.027e-9, 7.39e-9, 7.769e-9, 8.374e-9, 8.605e-9, 9.165e-9,
02306
02307
            9.415e-9, 9.511e-9, 9.704e-9, 9.588e-9, 9.45e-9, 9.086e-9,
02308
            8.798e-9, 8.469e-9, 7.697e-9, 7.168e-9, 6.255e-9, 5.772e-9,
            4.97e-9, 4.271e-9, 3.653e-9, 3.154e-9, 2.742e-9, 2.435e-9, 2.166e-9, 1.936e-9, 1.731e-9, 1.556e-9, 1.399e-9, 1.272e-9,
02309
02310
```

```
1.157e-9, 1.066e-9, 9.844e-10, 9.258e-10, 8.787e-10, 8.421e-10,
            8.083e-10, 8.046e-10, 8.067e-10, 8.181e-10, 8.325e-10, 8.517e-10, 9.151e-10, 9.351e-10, 9.677e-10, 1.071e-9, 1.126e-9, 1.219e-9,
02312
02313
02314
            1.297 e^{-9}, \ 1.408 e^{-9}, \ 1.476 e^{-9}, \ 1.517 e^{-9}, \ 1.6 e^{-9}, \ 1.649 e^{-9},
            1.678e-9, 1.746e-9, 1.742e-9, 1.728e-9, 1.699e-9, 1.655e-9 1.561e-9, 1.48e-9, 1.451e-9, 1.411e-9, 1.171e-9, 1.106e-9,
02315
02316
            9.714e-10, 8.523e-10, 7.346e-10, 6.241e-10, 5.371e-10, 4.704e-10,
            4.144e-10, 3.683e-10, 3.292e-10, 2.942e-10, 2.62e-10, 2.341e-10,
02318
            2.104e-10, 1.884e-10, 1.7e-10, 1.546e-10, 1.394e-10, 1.265e-10,
02319
02320
            1.14e-10, 1.019e-10, 9.279e-11, 8.283e-11, 7.458e-11, 6.668e-11,
            5.976e-11, 5.33e-11, 4.794e-11, 4.289e-11, 3.841e-11, 3.467e-11,
02321
02322
            3.13e-11, 2.832e-11, 2.582e-11, 2.356e-11, 2.152e-11, 1.97e-11,
            1.808e-11, 1.664e-11, 1.539e-11, 1.434e-11, 1.344e-11, 1.269e-11,
02323
            1.209e-11, 1.162e-11, 1.129e-11, 1.108e-11, 1.099e-11, 1.103e-11,
02324
02325
            1.119e-11, 1.148e-11, 1.193e-11, 1.252e-11, 1.329e-11, 1.421e-11,
            1.555e-11, 1.685e-11, 1.839e-11, 2.054e-11, 2.317e-11, 2.571e-11, 2.839e-11, 3.171e-11, 3.49e-11, 3.886e-11, 4.287e-11, 4.645e-11,
02326
02327
            5.047e-11, 5.592e-11, 6.109e-11, 6.628e-11, 7.381e-11, 8.088e-11,
02328
            8.966e-11, 1.045e-10, 1.12e-10, 1.287e-10, 1.486e-10, 1.662e-10,
02330
            1.866e-10, 2.133e-10, 2.524e-10, 2.776e-10, 3.204e-10, 3.559e-10,
            4.028e-10, 4.448e-10, 4.882e-10, 5.244e-10, 5.605e-10, 6.018e-10,
02331
02332
            6.328e-10, 6.579e-10, 6.541e-10, 7.024e-10, 7.074e-10, 7.068e-10,
02333
            7.009e-10, 6.698e-10, 6.545e-10, 6.209e-10, 5.834e-10, 5.412e-10,
            5.001e-10, 4.231e-10, 3.727e-10, 3.211e-10, 2.833e-10, 2.447e-10,
02334
            2.097e-10, 1.843e-10, 1.639e-10, 1.449e-10, 1.27e-10, 1.161e-10,
02335
            1.033e-10, 9.82e-11, 8.407e-11, 7.639e-11, 7.023e-11, 6.474e-11, 6.142e-11, 5.76e-11, 5.568e-11, 5.472e-11, 5.39e-11, 5.455e-11,
02336
02337
02338
            5.54e-11, 5.587e-11, 6.23e-11, 6.49e-11, 6.868e-11, 7.382e-11,
02339
            8.022e-11, 8.372e-11, 9.243e-11, 1.004e-10, 1.062e-10, 1.13e-10,
02340
            1.176e-10, 1.244e-10, 1.279e-10, 1.298e-10, 1.302e-10, 1.312e-10,
02341
            1.295e-10, 1.244e-10, 1.211e-10, 1.167e-10, 1.098e-10, 9.927e-11,
02342
            8.854e-11, 8.011e-11, 7.182e-11, 5.923e-11, 5.212e-11, 4.453e-11,
02343
            3.832e-11, 3.371e-11, 2.987e-11, 2.651e-11, 2.354e-11, 2.093e-11,
02344
            1.863e-11, 1.662e-11, 1.486e-11, 1.331e-11, 1.193e-11, 1.071e-11,
            9.628e-12, 8.66e-12, 7.801e-12, 7.031e-12, 6.347e-12, 5.733e-12, 5.182e-12, 4.695e-12, 4.26e-12, 3.874e-12, 3.533e-12, 3.235e-12, 2.979e-12, 2.76e-12, 2.579e-12, 2.432e-12, 2.321e-12, 2.246e-12, 2.205e-12, 2.196e-12, 2.223e-12, 2.288e-12, 2.387e-12, 2.525e-12,
02345
02346
02347
02349
            2.704e-12, 2.925e-12, 3.191e-12, 3.508e-12, 3.876e-12, 4.303e-12,
            4.793e-12, 5.347e-12, 5.978e-12, 6.682e-12, 7.467e-12, 8.34e-12,
02350
02351
            9.293e-12, 1.035e-11, 1.152e-11, 1.285e-11, 1.428e-11, 1.586e-11,
            1.764e-11, 1.972e-11, 2.214e-11, 2.478e-11, 2.776e-11, 3.151e-11, 3.591e-11, 4.103e-11, 4.66e-11, 5.395e-11, 6.306e-11, 7.172e-11, 8.358e-11, 9.67e-11, 1.11e-10, 1.325e-10, 1.494e-10, 1.736e-10,
02352
02353
02354
            2.007e-10, 2.296e-10, 2.608e-10, 3.004e-10, 3.361e-10, 3.727e-10,
02355
02356
            4.373e-10, 4.838e-10, 5.483e-10, 6.006e-10, 6.535e-10, 6.899e-10,
02357
            7.687e-10, 8.444e-10, 8.798e-10, 9.135e-10, 9.532e-10, 9.757e-10,
            9.968e-10, 1.006e-9, 9.949e-10, 9.789e-10, 9.564e-10, 9.215e-10, 8.51e-10, 8.394e-10, 7.707e-10, 7.152e-10, 6.274e-10, 5.598e-10,
02358
02359
            5.028e-10, 4.3e-10, 3.71e-10, 3.245e-10, 2.809e-10, 2.461e-10,
02360
            2.154e-10, 1.91e-10, 1.685e-10, 1.487e-10, 1.313e-10, 1.163e-10,
            1.031e-10, 9.172e-11, 8.221e-11, 7.382e-11, 6.693e-11, 6.079e-11,
02362
02363
            5.581e-11, 5.167e-11, 4.811e-11, 4.506e-11, 4.255e-11, 4.083e-11,
            3.949e-11, 3.881e-11, 3.861e-11, 3.858e-11, 3.951e-11, 4.045e-11, 4.24e-11, 4.487e-11, 4.806e-11, 5.133e-11, 5.518e-11, 5.919e-11,
02364
02365
            6.533e-11, 7.031e-11, 7.762e-11, 8.305e-11, 9.252e-11, 9.727e-11,
02366
            1.045e-10, 1.117e-10, 1.2e-10, 1.275e-10, 1.341e-10, 1.362e-10,
            1.438e-10, 1.45e-10, 1.455e-10, 1.455e-10, 1.434e-10, 1.381e-10,
02368
            1.301e-10, 1.276e-10, 1.163e-10, 1.089e-10, 9.911e-11, 8.943e-11,
02369
02370
            7.618e-11, 6.424e-11, 5.717e-11, 4.866e-11, 4.257e-11, 3.773e-11,
            3.331e-11, 2.958e-11, 2.629e-11, 2.316e-11, 2.073e-11, 1.841e-11, 1.635e-11, 1.464e-11, 1.31e-11, 1.16e-11, 1.047e-11, 9.408e-12, 8.414e-12, 7.521e-12, 6.705e-12, 5.993e-12, 5.371e-12, 4.815e-12,
02371
02372
            4.338e-12, 3.921e-12, 3.567e-12, 3.265e-12, 3.01e-12, 2.795e-12,
02374
02375
            2.613e-12, 2.464e-12, 2.346e-12, 2.256e-12, 2.195e-12, 2.165e-12,
02376
            2.166e-12, 2.198e-12, 2.262e-12, 2.364e-12, 2.502e-12, 2.682e-12,
02377
            2.908e-12, 3.187e-12, 3.533e-12, 3.946e-12, 4.418e-12, 5.013e-12, 5.708e-12, 6.379e-12, 7.43e-12, 8.39e-12, 9.51e-12, 1.078e-11,
02378
            1.259e-11, 1.438e-11, 1.63e-11, 1.814e-11, 2.055e-11, 2.348e-11, 2.664e-11, 2.956e-11, 3.3e-11, 3.677e-11, 4.032e-11, 4.494e-11,
02379
            4.951e-11, 5.452e-11, 6.014e-11, 6.5e-11, 6.915e-11, 7.45e-11,
02381
02382
            7.971e-11, 8.468e-11, 8.726e-11, 8.995e-11, 9.182e-11, 9.509e-11,
            9.333e-11, 9.386e-11, 9.457e-11, 9.21e-11, 9.019e-11, 8.68e-11, 8.298e-11, 7.947e-11, 7.46e-11, 7.082e-11, 6.132e-11, 5.855e-11,
02383
02384
            5.073e-11, 4.464e-11, 3.825e-11, 3.375e-11, 2.911e-11, 2.535e-11,
02385
            2.16e-11, 1.907e-11, 1.665e-11, 1.463e-11, 1.291e-11, 1.133e-11,
02386
            9.997e-12, 8.836e-12, 7.839e-12, 6.943e-12, 6.254e-12, 5.6e-12,
02387
02388
            5.029e-12, 4.529e-12, 4.102e-12, 3.737e-12, 3.428e-12, 3.169e-12,
02389
            2.959e-12, 2.798e-12, 2.675e-12, 2.582e-12, 2.644e-12, 2.557e-12,
            2.614e-12, 2.717e-12, 2.874e-12, 3.056e-12, 3.187e-12, 3.631e-12,
02390
            3.979e-12, 4.248e-12, 4.817e-12, 5.266e-12, 5.836e-12, 6.365e-12, 6.807e-12, 7.47e-12, 7.951e-12, 8.636e-12, 8.972e-12, 9.314e-12,
02391
            9.445e-12, 1.003e-11, 1.013e-11, 9.937e-12, 9.729e-12, 9.064e-12,
02393
02394
            9.119e-12, 9.124e-12, 8.704e-12, 8.078e-12, 7.47e-12, 6.329e-12,
02395
            5.674e-12, 4.808e-12, 4.119e-12, 3.554e-12, 3.103e-12, 2.731e-12,
            2.415e-12, 2.15e-12, 1.926e-12, 1.737e-12, 1.578e-12, 1.447e-12, 1.34e-12, 1.255e-12, 1.191e-12, 1.146e-12, 1.121e-12, 1.114e-12,
02396
02397
```

```
1.126e-12, 1.156e-12, 1.207e-12, 1.278e-12, 1.372e-12, 1.49e-12,
             1.633e-12, 1.805e-12, 2.01e-12, 2.249e-12, 2.528e-12, 2.852e-12, 3.228e-12, 3.658e-12, 4.153e-12, 4.728e-12, 5.394e-12, 6.176e-12,
02399
02400
             7.126e-12, 8.188e-12, 9.328e-12, 1.103e-11, 1.276e-11, 1.417e-11, 1.615e-11, 1.84e-11, 2.155e-11, 2.429e-11, 2.826e-11, 3.222e-11, 3.664e-11, 4.14e-11, 4.906e-11, 5.536e-11, 6.327e-11, 7.088e-11,
02401
02402
02403
              8.316e-11, 9.242e-11, 1.07e-10, 1.223e-10, 1.341e-10, 1.553e-10,
              1.703e-10, 1.9e-10, 2.022e-10, 2.233e-10, 2.345e-10, 2.438e-10,
02405
02406
              2.546e-10, 2.599e-10, 2.661e-10, 2.703e-10, 2.686e-10, 2.662e-10,
             2.56e-10, 2.552e-10, 2.378e-10, 2.252e-10, 2.146e-10, 1.885e-10, 1.668e-10, 1.441e-10, 1.295e-10, 1.119e-10, 9.893e-11, 8.687e-11, 7.678e-11, 6.685e-11, 5.879e-11, 5.127e-11, 4.505e-11, 3.997e-11,
02407
02408
02409
02410
             3.511e-11
02411
02412
          static double h2ofrn[2001] = { .01095, .01126, .01205, .01322, .0143, .01506, .01548, .01534, .01486, .01373, .01262, .01134, .01001, .008702, .007475, .006481, .00548, .0046, .003833, .00311, .002543, .002049, .00168, .001374, .001046, 8.193e-4, 6.267e-4,
02413
02414
02415
              4.968e-4, 3.924e-4, 2.983e-4, 2.477e-4, 1.997e-4, 1.596e-4,
02417
              1.331e-4, 1.061e-4, 8.942e-5, 7.168e-5, 5.887e-5, 4.848e-5,
02418
02419
             3.817e-5, 3.17e-5, 2.579e-5, 2.162e-5, 1.768e-5, 1.49e-5,
             1.231e-5, 1.013e-5, 8.555e-6, 7.328e-6, 6.148e-6, 5.207e-6, 4.387e-6, 3.741e-6, 3.22e-6, 2.753e-6, 2.346e-6, 1.985e-6, 1.716e-6, 1.475e-6, 1.286e-6, 1.122e-6, 9.661e-7, 8.284e-7,
02420
02421
02422
             7.057e-7, 6.119e-7, 5.29e-7, 4.571e-7, 3.948e-7, 3.432e-7, 2.983e-7, 2.589e-7, 2.265e-7, 1.976e-7, 1.704e-7, 1.456e-7,
02424
             1.26e-7, 1.101e-7, 9.648e-8, 8.415e-8, 7.34e-8, 6.441e-8, 5.643e-8, 4.94e-8, 4.276e-8, 3.703e-8, 3.227e-8, 2.825e-8,
02/25
02426
             2.478e-8, 2.174e-8, 1.898e-8, 1.664e-8, 1.458e-8, 1.278e-8,
02427
             1.126e-8, 9.891e-9, 8.709e-9, 7.652e-9, 6.759e-9, 5.975e-9,
02428
02429
              5.31e-9, 4.728e-9, 4.214e-9, 3.792e-9, 3.463e-9, 3.226e-9,
02430
              2.992e-9, 2.813e-9, 2.749e-9, 2.809e-9, 2.913e-9, 3.037e-9,
02431
             3.413e-9, 3.738e-9, 4.189e-9, 4.808e-9, 5.978e-9, 7.088e-9,
             8.071e-9, 9.61e-9, 1.21e-8, 1.5e-8, 1.764e-8, 2.221e-8, 2.898e-8, 3.948e-8, 5.068e-8, 6.227e-8, 7.898e-8, 1.033e-7, 1.437e-7, 1.889e-7, 2.589e-7, 3.59e-7, 4.971e-7, 7.156e-7, 9.983e-7,
02432
02433
02434
              1.381e-6, 1.929e-6, 2.591e-6, 3.453e-6, 4.57e-6, 5.93e-6,
02436
              7.552e-6, 9.556e-6, 1.183e-5, 1.425e-5, 1.681e-5, 1.978e-5,
             2.335e-5, 2.668e-5, 3.022e-5, 3.371e-5, 3.715e-5, 3.967e-5,
02437
02438
             4.06e-5, 4.01e-5, 3.809e-5, 3.491e-5, 3.155e-5, 2.848e-5,
             2.678e-5, 2.66e-5, 2.811e-5, 3.071e-5, 3.294e-5, 3.459e-5,
02439
             3.569e-5, 3.56e-5, 3.434e-5, 3.186e-5, 2.916e-5, 2.622e-5, 2.275e-5, 1.918e-5, 1.62e-5, 1.373e-5, 1.182e-5, 1.006e-5,
02440
02441
              8.556e-6, 7.26e-6, 6.107e-6, 5.034e-6, 4.211e-6, 3.426e-6,
02442
02443
             2.865e-6, 2.446e-6, 1.998e-6, 1.628e-6, 1.242e-6, 1.005e-6,
             7.853e-7, 6.21e-7, 5.071e-7, 4.156e-7, 3.548e-7, 2.825e-7, 2.261e-7, 1.916e-7, 1.51e-7, 1.279e-7, 1.059e-7, 9.14e-8, 7.707e-8, 6.17e-8, 5.311e-8, 4.263e-8, 3.518e-8, 2.961e-8,
02444
02445
02446
             2.457e-8, 2.119e-8, 1.712e-8, 1.439e-8, 1.201e-8, 1.003e-8,
02447
             8.564e-9, 7.199e-9, 6.184e-9, 5.206e-9, 4.376e-9, 3.708e-9,
             3.157e-9, 2.725e-9, 2.361e-9, 2.074e-9, 1.797e-9, 1.562e-9,
02449
02450
             1.364e-9, 1.196e-9, 1.042e-9, 8.862e-10, 7.648e-10, 6.544e-10,
             5.609e-10, 4.791e-10, 4.108e-10, 3.531e-10, 3.038e-10, 2.618e-10, 2.268e-10, 1.969e-10, 1.715e-10, 1.496e-10, 1.308e-10, 1.147e-10,
02451
02452
             1.008e-10, 8.894e-11, 7.885e-11, 7.031e-11, 6.355e-11, 5.854e-11, 5.534e-11, 5.466e-11, 5.725e-11, 6.447e-11, 7.943e-11, 1.038e-10,
02453
              1.437e-10, 2.04e-10, 2.901e-10, 4.051e-10, 5.556e-10, 7.314e-10,
02455
02456
             9.291e-10, 1.134e-9, 1.321e-9, 1.482e-9, 1.596e-9, 1.669e-9,
02457
             1.715e-9, 1.762e-9, 1.817e-9, 1.828e-9, 1.848e-9, 1.873e-9,
             1.902e-9, 1.894e-9, 1.864e-9, 1.841e-9, 1.797e-9, 1.704e-9, 1.559e-9, 1.382e-9, 1.187e-9, 1.001e-9, 8.468e-10, 7.265e-10,
02458
02459
              6.521e-10, 6.381e-10, 6.66e-10, 7.637e-10, 9.705e-10, 1.368e-9,
             1.856e-9, 2.656e-9, 3.954e-9, 5.96e-9, 8.72e-9, 1.247e-8,
02461
02462
              1.781e-8, 2.491e-8, 3.311e-8, 4.272e-8, 5.205e-8, 6.268e-8,
02463
             7.337e-8, 8.277e-8, 9.185e-8, 1.004e-7, 1.091e-7, 1.159e-7,
02464
             1.188e-7, 1.175e-7, 1.124e-7, 1.033e-7, 9.381e-8, 8.501e-8, 7.956e-8, 7.894e-8, 8.331e-8, 9.102e-8, 9.836e-8, 1.035e-7,
02465
              1.064e-7, 1.06e-7, 1.032e-7, 9.808e-8, 9.139e-8, 8.442e-8,
02466
              7.641e-8, 6.881e-8, 6.161e-8, 5.404e-8, 4.804e-8, 4.446e-8,
02468
              4.328e-8, 4.259e-8, 4.421e-8, 4.673e-8, 4.985e-8, 5.335e-8,
02469
             5.796e-8, 6.542e-8, 7.714e-8, 8.827e-8, 1.04e-7, 1.238e-7,
             1.499e-7, 1.829e-7, 2.222e-7, 2.689e-7, 3.303e-7, 3.981e-7, 4.84e-7, 5.91e-7, 7.363e-7, 9.087e-7, 1.139e-6, 1.455e-6,
02470
02471
              1.866e-6, 2.44e-6, 3.115e-6, 3.941e-6, 4.891e-6, 5.992e-6, 7.111e-6, 8.296e-6, 9.21e-6, 9.987e-6, 1.044e-5, 1.073e-5,
02472
02473
              1.092e-5, 1.106e-5, 1.138e-5, 1.171e-5, 1.186e-5, 1.186e-5,
02474
02475
              1.179e-5, 1.166e-5, 1.151e-5, 1.16e-5, 1.197e-5, 1.241e-5,
02476
              1.268e-5, 1.26e-5, 1.184e-5, 1.063e-5, 9.204e-6, 7.584e-6,
              6.053e-6, 4.482e-6, 3.252e-6, 2.337e-6, 1.662e-6, 1.18e-6,
02477
02478
             8.15e-7, 5.95e-7, 4.354e-7, 3.302e-7, 2.494e-7, 1.93e-7, 1.545e-7, 1.25e-7, 1.039e-7, 8.602e-8, 7.127e-8, 5.897e-8,
              4.838e-8, 4.018e-8, 3.28e-8, 2.72e-8, 2.307e-8, 1.972e-8,
02480
02481
             1.654e-8, 1.421e-8, 1.174e-8, 1.004e-8, 8.739e-9, 7.358e-9,
02482
              6.242e-9, 5.303e-9, 4.567e-9, 3.94e-9, 3.375e-9, 2.864e-9,
             2.422e-9, 2.057e-9, 1.75e-9, 1.505e-9, 1.294e-9, 1.101e-9, 9.401e-10, 8.018e-10, 6.903e-10, 5.965e-10, 5.087e-10, 4.364e-10,
02483
02484
```

```
3.759e-10, 3.247e-10, 2.809e-10, 2.438e-10, 2.123e-10, 1.853e-10,
             1.622e-10, 1.426e-10, 1.26e-10, 1.125e-10, 1.022e-10, 9.582e-11, 9.388e-11, 9.801e-11, 1.08e-10, 1.276e-10, 1.551e-10, 1.903e-10,
02486
02487
02488
             2.291 e^{-10},\ 2.724 e^{-10},\ 3.117 e^{-10},\ 3.4 e^{-10},\ 3.562 e^{-10},\ 3.625 e^{-10},
             3.619e-10, 3.429e-10, 3.221e-10, 2.943e-10, 2.645e-10, 2.338e-10, 2.062e-10, 1.901e-10, 1.814e-10, 1.827e-10, 1.906e-10, 1.984e-10,
02489
02490
             2.04e-10, 2.068e-10, 2.075e-10, 2.018e-10, 1.959e-10, 1.897e-10,
             1.852e-10, 1.791e-10, 1.696e-10, 1.634e-10, 1.598e-10, 1.561e-10,
02492
02493
             1.518e-10, 1.443e-10, 1.377e-10, 1.346e-10, 1.342e-10, 1.375e-10,
02494
             1.525e-10, 1.767e-10, 2.108e-10, 2.524e-10, 2.981e-10, 3.477e-10,
             4.262e-10, 5.326e-10, 6.646e-10, 8.321e-10, 1.069e-9, 1.386e-9,
02495
             1.743e-9, 2.216e-9, 2.808e-9, 3.585e-9, 4.552e-9, 5.907e-9,
02496
             7.611e-9, 9.774e-9, 1.255e-8, 1.666e-8, 2.279e-8, 3.221e-8,
02497
             4.531e-8, 6.4e-8, 9.187e-8, 1.295e-7, 1.825e-7, 2.431e-7,
02498
             3.181e-7, 4.009e-7, 4.941e-7, 5.88e-7, 6.623e-7, 7.155e-7
02499
             7.451e-7, 7.594e-7, 7.541e-7, 7.467e-7, 7.527e-7, 7.935e-7, 8.461e-7, 8.954e-7, 9.364e-7, 9.843e-7, 1.024e-6, 1.05e-6,
02500
02501
             1.059e-6, 1.074e-6, 1.072e-6, 1.043e-6, 9.789e-7, 8.803e-7, 7.662e-7, 6.378e-7, 5.133e-7, 3.958e-7, 2.914e-7, 2.144e-7,
02502
             1.57e-7, 1.14e-7, 8.47e-8, 6.2e-8, 4.657e-8, 3.559e-8, 2.813e-8,
             2.222e-8, 1.769e-8, 1.391e-8, 1.125e-8, 9.186e-9, 7.704e-9,
02505
02506
             6.447e-9, 5.381e-9, 4.442e-9, 3.669e-9, 3.057e-9, 2.564e-9,
             2.153e-9, 1.784e-9, 1.499e-9, 1.281e-9, 1.082e-9, 9.304e-10, 8.169e-10, 6.856e-10, 5.866e-10, 5.043e-10, 4.336e-10, 3.731e-10, 3.175e-10, 2.745e-10, 2.374e-10, 2.007e-10, 1.737e-10, 1.508e-10,
02507
02508
02509
             1.302e-10, 1.13e-10, 9.672e-11, 8.375e-11, 7.265e-11, 6.244e-11,
             5.343e-11, 4.654e-11, 3.975e-11, 3.488e-11, 3.097e-11, 2.834e-11,
02511
02512
             2.649e-11, 2.519e-11, 2.462e-11, 2.443e-11, 2.44e-11, 2.398e-11,
02513
             2.306e-11, 2.183e-11, 2.021e-11, 1.821e-11, 1.599e-11, 1.403e-11,
02514
             1.196e-11, 1.023e-11, 8.728e-12, 7.606e-12, 6.941e-12, 6.545e-12,
             6.484e-12, 6.6e-12, 6.718e-12, 6.785e-12, 6.746e-12, 6.724e-12,
02515
             6.764e-12, 6.995e-12, 7.144e-12, 7.32e-12, 7.33e-12, 7.208e-12,
             6.789e-12, 6.09e-12, 5.337e-12, 4.62e-12, 4.037e-12, 3.574e-12,
02517
02518
             3.311e-12, 3.346e-12, 3.566e-12, 3.836e-12, 4.076e-12, 4.351e-12,
            4.691e-12, 5.114e-12, 5.427e-12, 6.167e-12, 7.436e-12, 8.842e-12, 1.038e-11, 1.249e-11, 1.54e-11, 1.915e-11, 2.48e-11, 3.256e-11, 4.339e-11, 5.611e-11, 7.519e-11, 1.037e-10, 1.409e-10, 1.883e-10, 2.503e-10, 3.38e-10, 4.468e-10, 5.801e-10, 7.335e-10, 8.98e-10,
02519
02520
02521
02523
             1.11e-9, 1.363e-9, 1.677e-9, 2.104e-9, 2.681e-9, 3.531e-9,
             4.621e-9, 6.106e-9, 8.154e-9, 1.046e-8, 1.312e-8, 1.607e-8
02524
02525
             1.948e-8, 2.266e-8, 2.495e-8, 2.655e-8, 2.739e-8, 2.739e-8,
             2.662e-8, 2.589e-8, 2.59e-8, 2.664e-8, 2.833e-8, 3.023e-8,
02526
            3.305e-8, 3.558e-8, 3.793e-8, 3.961e-8, 4.056e-8, 4.102e-8, 4.025e-8, 3.917e-8, 3.706e-8, 3.493e-8, 3.249e-8, 3.096e-8,
02527
             3.011e-8, 3.111e-8, 3.395e-8, 3.958e-8, 4.875e-8, 6.066e-8,
02529
02530
             7.915e-8, 1.011e-7, 1.3e-7, 1.622e-7, 2.003e-7, 2.448e-7,
            2.863e-7, 3.317e-7, 3.655e-7, 3.96e-7, 4.098e-7, 4.168e-7, 4.198e-7, 4.207e-7, 4.289e-7, 4.384e-7, 4.471e-7, 4.524e-7, 4.574e-7, 4.633e-7, 4.785e-7, 5.028e-7, 5.371e-7, 5.727e-7,
02531
02532
02533
             5.955e-7, 5.998e-7, 5.669e-7, 5.082e-7, 4.397e-7, 3.596e-7,
02534
             2.814e-7, 2.074e-7, 1.486e-7, 1.057e-7, 7.25e-8, 4.946e-8,
             3.43e-8, 2.447e-8, 1.793e-8, 1.375e-8, 1.096e-8, 9.091e-9,
02536
02537
             7.709e-9, 6.631e-9, 5.714e-9, 4.886e-9, 4.205e-9, 3.575e-9
            3.07e-9, 2.631e-9, 2.284e-9, 2.002e-9, 1.745e-9, 1.509e-9, 1.284e-9, 1.084e-9, 9.163e-10, 7.663e-10, 6.346e-10, 5.283e-10,
02538
02539
             4.354e-10, 3.59e-10, 2.982e-10, 2.455e-10, 2.033e-10, 1.696e-10, 1.432e-10, 1.211e-10, 1.02e-10, 8.702e-11, 7.38e-11, 6.293e-11,
02540
             5.343e-11, 4.532e-11, 3.907e-11, 3.365e-11, 2.945e-11, 2.558e-11,
02542
            3.343e-11, 4.332e-11, 3.307e-11, 3.305e-11, 2.336e-11, 2.336e-11, 2.192e-11, 1.895e-11, 1.636e-11, 1.42e-11, 1.228e-11, 1.063e-11, 9.348e-12, 8.2e-12, 7.231e-12, 6.43e-12, 5.702e-12, 5.052e-12, 4.469e-12, 4e-12, 3.679e-12, 3.387e-12, 3.197e-12, 3.158e-12, 3.327e-12, 3.675e-12, 4.292e-12, 5.437e-12, 7.197e-12, 1.008e-11, 1.437e-11, 2.035e-11, 2.905e-11, 4.062e-11, 5.528e-11, 7.177e-11,
02543
02544
02545
02546
02548
             9.064e-11, 1.109e-10, 1.297e-10, 1.473e-10, 1.652e-10, 1.851e-10,
02549
             2.079e-10, 2.313e-10, 2.619e-10, 2.958e-10, 3.352e-10, 3.796e-10,
02550
             4.295e-10, 4.923e-10, 5.49e-10, 5.998e-10, 6.388e-10, 6.645e-10,
02551
             6.712e-10, 6.549e-10, 6.38e-10, 6.255e-10, 6.253e-10, 6.459e-10, 6.977e-10, 7.59e-10, 8.242e-10, 8.92e-10, 9.403e-10, 9.701e-10,
02552
02553
             9.483e-10, 9.135e-10, 8.617e-10, 7.921e-10, 7.168e-10, 6.382e-10,
             5.677e-10, 5.045e-10, 4.572e-10, 4.312e-10, 4.145e-10, 4.192e-10,
             4.541e-10, 5.368e-10, 6.771e-10, 8.962e-10, 1.21e-9, 1.659e-9,
02555
02556
             2.33e-9, 3.249e-9, 4.495e-9, 5.923e-9, 7.642e-9, 9.607e-9,
            1.178e-8, 1.399e-8, 1.584e-8, 1.73e-8, 1.816e-8, 1.87e-8, 1.868e-8, 1.87e-8, 1.884e-8, 1.99e-8, 2.15e-8, 2.258e-8,
02557
02558
             2.364e-8, 2.473e-8, 2.602e-8, 2.689e-8, 2.731e-8, 2.816e-8,
02559
             2.859e-8, 2.839e-8, 2.703e-8, 2.451e-8, 2.149e-8, 1.787e-8,
02560
             1.449e-8, 1.111e-8, 8.282e-9, 6.121e-9, 4.494e-9, 3.367e-9,
02561
02562
             2.487e-9, 1.885e-9, 1.503e-9, 1.249e-9, 1.074e-9, 9.427e-10,
             8.439e-10, 7.563e-10, 6.772e-10, 6.002e-10, 5.254e-10, 4.588e-10,
02563
             3.977e-10, 3.449e-10, 3.003e-10, 2.624e-10, 2.335e-10, 2.04e-10,
02564
             1.771e-10, 1.534e-10, 1.296e-10, 1.097e-10, 9.173e-11, 7.73e-11,
02565
             6.547e-11, 5.191e-11, 4.198e-11, 3.361e-11, 2.732e-11, 2.244e-11,
             1.791e-11, 1.509e-11, 1.243e-11, 1.035e-11, 8.969e-12, 7.394e-12,
02567
02568
             6.323e-12, 5.282e-12, 4.543e-12, 3.752e-12, 3.14e-12, 2.6e-12,
02569
             2.194e-12, 1.825e-12, 1.511e-12, 1.245e-12, 1.024e-12, 8.539e-13,
             7.227e-13, 6.102e-13, 5.189e-13, 4.43e-13, 3.774e-13, 3.236e-13, 2.8e-13, 2.444e-13, 2.156e-13, 1.932e-13, 1.775e-13, 1.695e-13,
02570
02571
```

```
1.672e-13, 1.704e-13, 1.825e-13, 2.087e-13, 2.614e-13, 3.377e-13,
            4.817e-13, 6.89e-13, 1.062e-12, 1.562e-12, 2.288e-12, 3.295e-12, 4.55e-12, 5.965e-12, 7.546e-12, 9.395e-12, 1.103e-11, 1.228e-11,
02573
02574
02575
            1.318e-11,\ 1.38e-11,\ 1.421e-11,\ 1.39e-11,\ 1.358e-11,\ 1.336e-11,
02576
            1.342e-11, 1.356e-11, 1.424e-11, 1.552e-11, 1.73e-11, 1.951e-11,
            2.128e-11, 2.249e-11, 2.277e-11, 2.226e-11, 2.111e-11, 1.922e-11,
            1.775e-11, 1.661e-11, 1.547e-11, 1.446e-11, 1.323e-11, 1.21e-11,
            1.054e-11, 9.283e-12, 8.671e-12, 8.67e-12, 9.429e-12, 1.062e-11,
02579
02580
            1.255e-11, 1.506e-11, 1.818e-11, 2.26e-11, 2.831e-11, 3.723e-11,
            5.092e-11, 6.968e-11, 9.826e-11, 1.349e-10, 1.87e-10, 2.58e-10, 3.43e-10, 4.424e-10, 5.521e-10, 6.812e-10, 8.064e-10, 9.109e-10,
02581
02582
            9.839e-10, 1.028e-9, 1.044e-9, 1.029e-9, 1.005e-9, 1.002e-9, 1.038e-9, 1.122e-9, 1.233e-9, 1.372e-9, 1.524e-9, 1.665e-9,
02583
            1.804e-9, 1.908e-9, 2.015e-9, 2.117e-9, 2.219e-9, 2.336e-9,
02585
02586
            2.531e-9, 2.805e-9, 3.189e-9, 3.617e-9, 4.208e-9, 4.911e-9,
            5.619e-9, 6.469e-9, 7.188e-9, 7.957e-9, 8.503e-9, 9.028e-9, 9.571e-9, 9.99e-9, 1.055e-8, 1.102e-8, 1.132e-8, 1.141e-8,
02587
02588
            1.145e-8, 1.145e-8, 1.176e-8, 1.224e-8, 1.304e-8, 1.388e-8,
02589
            1.445e-8, 1.453e-8, 1.368e-8, 1.22e-8, 1.042e-8, 8.404e-9,
02591
            6.403e-9, 4.643e-9, 3.325e-9, 2.335e-9, 1.638e-9, 1.19e-9,
02592
            9.161e-10, 7.412e-10, 6.226e-10, 5.516e-10, 5.068e-10, 4.831e-10,
02593
            4.856e-10, 5.162e-10, 5.785e-10, 6.539e-10, 7.485e-10,
                                                                              8.565e-10.
            9.534e-10, 1.052e-9, 1.115e-9, 1.173e-9, 1.203e-9, 1.224e-9,
02594
            1.243e-9, 1.248e-9, 1.261e-9, 1.265e-9, 1.25e-9, 1.217e-9, 1.176e-9, 1.145e-9, 1.153e-9, 1.199e-9, 1.278e-9, 1.366e-9,
02595
02597
            1.426e-9, 1.444e-9, 1.365e-9, 1.224e-9, 1.051e-9, 8.539e-10,
02598
            6.564e-10, 4.751e-10, 3.404e-10, 2.377e-10, 1.631e-10, 1.114e-10,
02599
            7.87e-11, 5.793e-11, 4.284e-11, 3.3e-11, 2.62e-11, 2.152e-11,
02600
            1.777e-11, 1.496e-11, 1.242e-11, 1.037e-11, 8.725e-12, 7.004e-12,
            5.718e-12, 4.769e-12, 3.952e-12, 3.336e-12, 2.712e-12, 2.213e-12,
02601
            1.803e-12, 1.492e-12, 1.236e-12, 1.006e-12, 8.384e-13, 7.063e-13,
02602
02603
            5.879e-13, 4.93e-13, 4.171e-13, 3.569e-13, 3.083e-13, 2.688e-13,
            2.333e-13, 2.035e-13, 1.82e-13, 1.682e-13, 1.635e-13, 1.674e-13,
02604
02605
            1.769e-13, 2.022e-13, 2.485e-13, 3.127e-13, 4.25e-13, 5.928e-13,
02606
            8.514e-13, 1.236e-12, 1.701e-12, 2.392e-12, 3.231e-12, 4.35e-12,
            5.559e-12, 6.915e-12, 8.519e-12, 1.013e-11, 1.146e-11, 1.24e-11,
02607
02608
            1.305e-11, 1.333e-11, 1.318e-11, 1.263e-11, 1.238e-11, 1.244e-11,
            1.305e-11, 1.432e-11, 1.623e-11, 1.846e-11, 2.09e-11, 2.328e-11,
02610
            2.526e-11, 2.637e-11, 2.702e-11, 2.794e-11, 2.889e-11, 2.989e-11,
            3.231e-11, 3.68e-11, 4.375e-11, 5.504e-11, 7.159e-11, 9.502e-11,
02611
02612
            1.279 e^{-10}, \ 1.645 e^{-10}, \ 2.098 e^{-10}, \ 2.618 e^{-10}, \ 3.189 e^{-10}, \ 3.79 e^{-10}, \\
            4.303e-10, 4.753e-10, 5.027e-10, 5.221e-10, 5.293e-10, 5.346e-10,
02613
            5.467e-10, 5.796e-10, 6.2e-10, 6.454e-10, 6.705e-10, 6.925e-10, 7.233e-10, 7.35e-10, 7.538e-10, 7.861e-10, 8.077e-10, 8.132e-10,
02614
02615
            7.749e-10, 7.036e-10, 6.143e-10, 5.093e-10, 4.089e-10, 3.092e-10, 2.299e-10, 1.705e-10, 1.277e-10, 9.723e-11, 7.533e-11, 6.126e-11,
02616
02617
02618
            5.154e-11, 4.428e-11, 3.913e-11, 3.521e-11, 3.297e-11, 3.275e-11,
           3.46e-11, 3.798e-11, 4.251e-11, 4.745e-11, 5.232e-11, 5.606e-11, 5.82e-11, 5.88e-11, 5.79e-11, 5.661e-11, 5.491e-11, 5.366e-11, 5.341e-11, 5.353e-11, 5.336e-11, 5.293e-11, 5.248e-11, 5.235e-11,
02619
02620
02621
            5.208e-11, 5.322e-11, 5.521e-11, 5.725e-11, 5.827e-11, 5.685e-11,
            5.245e-11, 4.612e-11, 3.884e-11, 3.129e-11, 2.404e-11, 1.732e-11,
02623
02624
            1.223e-11, 8.574e-12, 5.888e-12, 3.986e-12, 2.732e-12, 1.948e-12,
02625
            1.414e-12, 1.061e-12, 8.298e-13, 6.612e-13, 5.413e-13, 4.472e-13,
02626
            3.772e-13, 3.181e-13, 2.645e-13, 2.171e-13, 1.778e-13, 1.464e-13,
            1.183e-13, 9.637e-14, 7.991e-14, 6.668e-14, 5.57e-14, 4.663e-14,
02627
            3.848e-14, 3.233e-14, 2.706e-14, 2.284e-14, 1.944e-14, 1.664e-14,
            1.43e-14, 1.233e-14, 1.066e-14, 9.234e-15, 8.023e-15, 6.993e-15,
02629
02630
            6.119e-15, 5.384e-15, 4.774e-15, 4.283e-15, 3.916e-15, 3.695e-15,
02631
            3.682e-15, 4.004e-15, 4.912e-15, 6.853e-15, 1.056e-14, 1.712e-14,
02632
            2.804e-14, 4.516e-14, 7.113e-14, 1.084e-13, 1.426e-13, 1.734e-13,
            1.978e-13, 2.194e-13, 2.388e-13, 2.489e-13, 2.626e-13, 2.865e-13,
02633
            3.105e-13, 3.387e-13, 3.652e-13, 3.984e-13, 4.398e-13, 4.906e-13,
            5.55e-13, 6.517e-13, 7.813e-13, 9.272e-13, 1.164e-12, 1.434e-12,
02635
02636
            1.849e-12, 2.524e-12, 3.328e-12, 4.523e-12, 6.108e-12, 8.207e-12,
02637
            1.122e-11, 1.477e-11, 1.9e-11, 2.412e-11, 2.984e-11, 3.68e-11,
02638
            4.353e-11, 4.963e-11, 5.478e-11, 5.903e-11, 6.233e-11, 6.483e-11, 6.904e-11, 7.569e-11, 8.719e-11, 1.048e-10, 1.278e-10, 1.557e-10,
02639
            1.869e-10, 2.218e-10, 2.61e-10, 2.975e-10, 3.371e-10, 3.746e-10,
02640
            4.065e-10, 4.336e-10, 4.503e-10, 4.701e-10, 4.8e-10, 4.917e-10,
            5.038e-10, 5.128e-10, 5.143e-10, 5.071e-10, 5.019e-10, 5.025e-10,
02642
02643
            5.183e-10, 5.496e-10, 5.877e-10, 6.235e-10, 6.42e-10, 6.234e-10,
            5.698e-10, 4.916e-10, 4.022e-10, 3.126e-10, 2.282e-10, 1.639e-10, 1.142e-10, 7.919e-11, 5.69e-11, 4.313e-11, 3.413e-11, 2.807e-11, 2.41e-11, 2.166e-11, 2.024e-11, 1.946e-11, 1.929e-11, 1.963e-11,
02644
02645
02646
            2.035e-11, 2.162e-11, 2.305e-11, 2.493e-11, 2.748e-11, 3.048e-11,
02647
            3.413e-11, 3.754e-11, 4.155e-11, 4.635e-11, 5.11e-11, 5.734e-11,
02648
            6.338e-11, 6.99e-11, 7.611e-11, 8.125e-11, 8.654e-11, 8.951e-11, 9.182e-11, 9.31e-11, 9.273e-11, 9.094e-11, 8.849e-11, 8.662e-11,
02649
02650
            8.67e-11, 8.972e-11, 9.566e-11, 1.025e-10, 1.083e-10, 1.111e-10,
02651
            1.074e-10, 9.771e-11, 8.468e-11, 6.958e-11, 5.47e-11, 4.04e-11, 2.94e-11, 2.075e-11, 1.442e-11, 1.01e-11, 7.281e-12, 5.409e-12,
02652
            4.138e-12, 3.304e-12, 2.784e-12, 2.473e-12, 2.273e-12, 2.186e-12,
02654
02655
            2.118e-12, 2.066e-12, 1.958e-12, 1.818e-12, 1.675e-12, 1.509e-12,
02656
            1.349e-12, 1.171e-12, 9.838e-13, 8.213e-13, 6.765e-13, 5.378e-13,
            4.161e-13, 3.119e-13, 2.279e-13, 1.637e-13, 1.152e-13, 8.112e-14, 5.919e-14, 4.47e-14, 3.492e-14, 2.811e-14, 2.319e-14, 1.948e-14,
02657
02658
```

```
1.66e-14, 1.432e-14, 1.251e-14, 1.109e-14, 1.006e-14, 9.45e-15,
            9.384e-15, 1.012e-14, 1.216e-14, 1.636e-14, 2.305e-14, 3.488e-14, 5.572e-14, 8.479e-14, 1.265e-13, 1.905e-13, 2.73e-13, 3.809e-13,
02660
02661
02662
            4.955e-13, 6.303e-13, 7.861e-13, 9.427e-13, 1.097e-12, 1.212e-12,
            1.328e-12, 1.415e-12, 1.463e-12, 1.495e-12, 1.571e-12, 1.731e-12, 1.981e-12, 2.387e-12, 2.93e-12, 3.642e-12, 4.584e-12, 5.822e-12,
02663
02664
            7.278e-12, 9.193e-12, 1.135e-11, 1.382e-11, 1.662e-11, 1.958e-11,
            2.286e-11, 2.559e-11, 2.805e-11, 2.988e-11, 3.106e-11,
            3.2e-11, 3.258e-11, 3.362e-11, 3.558e-11, 3.688e-11, 3.8e-11,
02667
02668
            3.929e-11, 4.062e-11, 4.186e-11, 4.293e-11, 4.48e-11, 4.643e-11,
            4.704 e^{-11},\ 4.571 e^{-11},\ 4.206 e^{-11},\ 3.715 e^{-11},\ 3.131 e^{-11},\ 2.541 e^{-11},
02669
02670
            1.978e-11, 1.508e-11, 1.146e-11, 8.7e-12, 6.603e-12, 5.162e-12,
            4.157e-12, 3.408e-12, 2.829e-12, 2.405e-12, 2.071e-12, 1.826e-12,
02671
            1.648e-12, 1.542e-12, 1.489e-12, 1.485e-12, 1.493e-12, 1.545e-12,
02672
02673
            1.637e-12, 1.814e-12, 2.061e-12, 2.312e-12, 2.651e-12, 3.03e-12,
           3.46e-12, 3.901e-12, 4.306e-12, 4.721e-12, 5.008e-12, 5.281e-12, 5.541e-12, 5.791e-12, 6.115e-12, 6.442e-12, 6.68e-12, 6.791e-12, 6.831e-12, 6.839e-12, 6.946e-12, 7.128e-12, 7.537e-12, 8.036e-12, 8.392e-12, 8.526e-12, 8.11e-12, 7.325e-12, 6.329e-12, 5.183e-12,
02674
02675
02676
            4.081e-12, 2.985e-12, 2.141e-12, 1.492e-12, 1.015e-12, 6.684e-13,
            4.414e-13, 2.987e-13, 2.038e-13, 1.391e-13, 9.86e-14, 7.24e-14,
02679
02680
            5.493e-14, 4.288e-14, 3.427e-14, 2.787e-14, 2.296e-14, 1.909e-14,
            1.598e-14, 1.344e-14, 1.135e-14, 9.616e-15, 8.169e-15, 6.957e-15, 5.938e-15, 5.08e-15, 4.353e-15, 3.738e-15, 3.217e-15, 2.773e-15, 2.397e-15, 2.077e-15, 1.805e-15, 1.575e-15, 1.382e-15, 1.221e-15,
02681
02682
02683
            1.09e-15, 9.855e-16, 9.068e-16, 8.537e-16, 8.27e-16, 8.29e-16,
            8.634e-16, 9.359e-16, 1.055e-15, 1.233e-15, 1.486e-15, 1.839e-15,
02685
02686
            2.326e-15, 2.998e-15, 3.934e-15, 5.256e-15, 7.164e-15, 9.984e-15,
02687
            1.427e-14, 2.099e-14, 3.196e-14, 5.121e-14, 7.908e-14, 1.131e-13,
02688
            1.602e-13, 2.239e-13, 3.075e-13, 4.134e-13, 5.749e-13, 7.886e-13,
            1.071e-12, 1.464e-12, 2.032e-12, 2.8e-12, 3.732e-12, 4.996e-12,
02689
            6.483e-12, 8.143e-12, 1.006e-11, 1.238e-11, 1.484e-11, 1.744e-11,
            2.02e-11, 2.274e-11, 2.562e-11, 2.848e-11, 3.191e-11, 3.617e-11,
02691
02692
            4.081e-11, 4.577e-11, 4.937e-11, 5.204e-11, 5.401e-11, 5.462e-11,
            5.507e-11, 5.51e-11, 5.605e-11, 5.686e-11, 5.739e-11, 5.766e-11, 5.74e-11, 5.754e-11, 5.761e-11, 5.777e-11, 5.712e-11, 5.51e-11,
02693
02694
            5.088e-11, 4.438e-11, 3.728e-11, 2.994e-11, 2.305e-11, 1.715e-11,
02695
            1.256e-11, 9.208e-12, 6.745e-12, 5.014e-12, 3.785e-12, 2.9e-12,
            2.239e-12, 1.757e-12, 1.414e-12, 1.142e-12, 9.482e-13, 8.01e-13,
            6.961e-13, 6.253e-13, 5.735e-13, 5.433e-13, 5.352e-13, 5.493e-13,
02698
02699
            5.706e-13, 6.068e-13, 6.531e-13, 7.109e-13, 7.767e-13, 8.59e-13,
            9.792e-13, 1.142e-12, 1.371e-12, 1.65e-12, 1.957e-12, 2.302e-12,
02700
02701
            2.705e-12, 3.145e-12, 3.608e-12, 4.071e-12, 4.602e-12, 5.133e-12,
02702
            5.572e-12, 5.987e-12, 6.248e-12, 6.533e-12, 6.757e-12, 6.935e-12,
            7.224e-12, 7.422e-12, 7.538e-12, 7.547e-12, 7.495e-12, 7.543e-12,
02703
02704
            7.725e-12, 8.139e-12, 8.627e-12, 9.146e-12, 9.443e-12, 9.318e-12,
02705
            8.649e-12, 7.512e-12, 6.261e-12, 4.915e-12, 3.647e-12, 2.597e-12,
           1.785e-12, 1.242e-12, 8.66e-13, 6.207e-13, 4.61e-13, 3.444e-13, 2.634e-13, 2.1e-13, 1.725e-13, 1.455e-13, 1.237e-13, 1.085e-13,
02706
02707
            9.513e-14, 7.978e-14, 6.603e-14, 5.288e-14, 4.084e-14, 2.952e-14,
02708
            2.157e-14, 1.593e-14, 1.199e-14, 9.267e-15, 7.365e-15, 6.004e-15,
            4.995e-15, 4.218e-15, 3.601e-15, 3.101e-15, 2.692e-15, 2.36e-15,
02710
02711
            2.094e-15, 1.891e-15, 1.755e-15, 1.699e-15, 1.755e-15, 1.987e-15,
02712
            2.506e-15, 3.506e-15, 5.289e-15, 8.311e-15, 1.325e-14, 2.129e-14,
02713
            3.237e-14, 4.595e-14, 6.441e-14, 8.433e-14, 1.074e-13, 1.383e-13,
            1.762e-13, 2.281e-13, 2.831e-13, 3.523e-13, 4.38e-13, 5.304e-13,
02714
            6.29e-13, 7.142e-13, 8.032e-13, 8.934e-13, 9.888e-13, 1.109e-12,
            1.261e-12, 1.462e-12, 1.74e-12, 2.099e-12, 2.535e-12, 3.008e-12,
02716
02717
            3.462e-12, 3.856e-12, 4.098e-12, 4.239e-12, 4.234e-12, 4.132e-12,
02718
            3.986e-12, 3.866e-12, 3.829e-12, 3.742e-12, 3.705e-12, 3.694e-12,
02719
            3.765e-12,\ 3.849e-12,\ 3.929e-12,\ 4.056e-12,\ 4.092e-12,\ 4.047e-12,
            3.792e-12, 3.407e-12, 2.953e-12, 2.429e-12, 1.931e-12, 1.46e-12,
02720
            1.099e-12, 8.199e-13, 6.077e-13, 4.449e-13, 3.359e-13, 2.524e-13,
            1.881e-13, 1.391e-13, 1.02e-13, 7.544e-14, 5.555e-14, 4.22e-14,
02722
02723
            3.321e-14, 2.686e-14, 2.212e-14, 1.78e-14, 1.369e-14, 1.094e-14,
02724
            9.13e-15, 8.101e-15, 7.828e-15, 8.393e-15, 1.012e-14, 1.259e-14,
           1.538e-14, 1.961e-14, 2.619e-14, 3.679e-14, 5.049e-14, 6.917e-14, 8.88e-14, 1.115e-13, 1.373e-13, 1.619e-13, 1.878e-13, 2.111e-13, 2.33e-13, 2.503e-13, 2.613e-13, 2.743e-13, 2.826e-13, 2.976e-13,
02725
02726
02727
            3.162e-13, 3.36e-13, 3.491e-13, 3.541e-13, 3.595e-13, 3.608e-13,
            3.709e-13, 3.869e-13, 4.12e-13, 4.366e-13, 4.504e-13, 4.379e-13,
02729
02730
            3.955e-13, 3.385e-13, 2.741e-13, 2.089e-13, 1.427e-13, 9.294e-14,
            5.775e-14, 3.565e-14, 2.21e-14, 1.398e-14, 9.194e-15, 6.363e-15, 4.644e-15, 3.55e-15, 2.808e-15, 2.274e-15, 1.871e-15, 1.557e-15,
02731
02732
02733
            1.308e-15, 1.108e-15, 9.488e-16, 8.222e-16, 7.238e-16, 6.506e-16,
            6.008e-16, 5.742e-16, 5.724e-16, 5.991e-16, 6.625e-16, 7.775e-16,
02734
            9.734e-16, 1.306e-15, 1.88e-15, 2.879e-15, 4.616e-15, 7.579e-15,
02735
02736
            1.248e-14, 2.03e-14, 3.244e-14, 5.171e-14, 7.394e-14, 9.676e-14,
            1.199e-13, 1.467e-13, 1.737e-13, 2.02e-13, 2.425e-13, 3.016e-13, 3.7e-13, 4.617e-13, 5.949e-13, 7.473e-13, 9.378e-13, 1.191e-12, 1.481e-12, 1.813e-12, 2.232e-12, 2.722e-12, 3.254e-12, 3.845e-12,
02737
02738
02739
            4.458e-12, 5.048e-12, 5.511e-12, 5.898e-12, 6.204e-12, 6.293e-12,
            6.386e-12, 6.467e-12, 6.507e-12, 6.466e-12, 6.443e-12, 6.598e-12, 6.873e-12, 7.3e-12, 7.816e-12, 8.368e-12, 8.643e-12, 8.466e-12,
02741
02742
02743
            7.871e-12, 6.853e-12, 5.714e-12, 4.482e-12, 3.392e-12, 2.613e-12,
            2.008e-12, 1.562e-12, 1.228e-12, 9.888e-13, 7.646e-13, 5.769e-13, 4.368e-13, 3.324e-13, 2.508e-13, 1.916e-13
02744
02745
```

```
02746
        };
02747
02748
        static double xfcrev[15] =
         { 1.003, 1.009, 1.015, 1.023, 1.029, 1.033, 1.037,
02749
02750
           1.039, 1.04, 1.046, 1.036, 1.027, 1.01, 1.002, 1.
02751
02752
02753
        double a1, a2, a3, dw, ew, dx, xw, xx, vf2, vf6, cw260, cw296,
02754
          sfac, fscal, cwfrn, ctmpth, ctwfrn, ctwslf;
02755
02756
        int iw, jw, ix;
02757
02758
        /* Get H2O continuum absorption... */
02759
        xw = nu / 10 + 1;
02760
        if (xw >= 1 && xw < 2001) {
02761
          iw = (int) xw;
          jw = iw + 1;

dw = xw - iw;
02762
02763
          ew = 1 - dw;
02764
          cw296 = ew * h2o296[iw - 1] + dw * h2o296[jw - 1];
cw260 = ew * h2o260[iw - 1] + dw * h2o260[jw - 1];
cwfrn = ew * h2ofrn[iw - 1] + dw * h2ofrn[jw - 1];
02765
02766
02767
           if (nu <= 820 || nu >= 960) {
02768
             sfac = 1;
02769
02770
           } else {
           xx = (nu - 820) / 10;
02771
02772
             ix = (int) xx;
02773
             dx = xx - ix;
02774
             sfac = (1 - dx) * xfcrev[ix] + dx * xfcrev[ix + 1];
02775
02776
          ctwslf = sfac * cw296 * pow(cw260 / cw296, (296 - t) / (296 - 260));
02777
           vf2 = gsl_pow_2(nu - 370);
02778
           vf6 = gsl_pow_3(vf2);
02779
           fscal = 36100 / (vf2 + vf6 * 1e-8 + 36100) * -.25 + 1;
           ctwfrn = cwfrn * fscal;
02780
          a1 = nu * u * tanh(.7193876 / t * nu);
a2 = 296 / t;
02781
02782
          a3 = p / P0 * (q * ctwslf + (1 - q) * ctwfrn) * 1e-20;
02784
          ctmpth = a1 * a2 * a3;
02785
02786
          ctmpth = 0;
02787
        return ctmpth;
02788 }
```

## 5.13.2.8 double ctmn2 ( double nu, double p, double t )

Compute nitrogen continuum (absorption coefficient).

Definition at line 2792 of file jurassic.c.

```
02795
02796
           static double ba[98] = { 0., 4.45e-8, 5.22e-8, 6.46e-8, 7.75e-8, 9.03e-8, 1.06e-7, 1.21e-7, 1.37e-7, 1.57e-7, 1.75e-7, 2.01e-7, 2.3e-7,
02797
02799
               2.59e-7, 2.95e-7, 3.26e-7, 3.66e-7, 4.05e-7, 4.47e-7, 4.92e-7,
02800
               5.34e-7, 5.84e-7, 6.24e-7, 6.67e-7, 7.14e-7, 7.26e-7, 7.54e-7,
02801
               7.84e-7, 8.09e-7, 8.42e-7, 8.62e-7, 8.87e-7, 9.11e-7, 9.36e-7,
02802
               9.76e-7, 1.03e-6, 1.11e-6, 1.23e-6, 1.39e-6, 1.61e-6, 1.76e-6,
               1.94e-6, 1.97e-6, 1.87e-6, 1.75e-6, 1.56e-6, 1.42e-6, 1.35e-6,
02803
02804
              1.32e-6, 1.29e-6, 1.29e-6, 1.3e-6, 1.3e-6, 1.32e-6, 1.33e-6,
               1.34e-6, 1.35e-6, 1.33e-6, 1.31e-6, 1.29e-6, 1.24e-6, 1.2e-6,
               1.16e-6, 1.1e-6, 1.04e-6, 9.96e-7, 9.38e-7, 8.63e-7, 7.98e-7, 7.26e-7, 6.55e-7, 5.94e-7, 5.35e-7, 4.74e-7, 4.24e-7, 3.77e-7,
02806
02807
              3.33e-7, 2.96e-7, 2.63e-7, 2.34e-7, 2.08e-7, 1.85e-7, 1.67e-7, 1.47e-7, 1.32e-7, 1.2e-7, 1.09e-7, 9.85e-8, 9.08e-8, 8.18e-8,
02808
02809
               7.56e-8, 6.85e-8, 6.14e-8, 5.83e-8, 5.77e-8, 5e-8, 4.32e-8, 0.
02810
02811
02812
           static double betaa[98] = { 802., 802., 761., 722., 679., 646., 609., 562.,
511., 472., 436., 406., 377., 355., 338., 319., 299., 278., 255.,
233., 208., 184., 149., 107., 66., 25., -13., -49., -82., -104.,
-119., -130., -139., -144., -146., -146., -147., -148., -150.,
-153., -160., -169., -181., -189., -195., -200., -205., -209.,
02813
02814
02815
02816
02817
              -211, -210, -210, -209, -205, -199, -190, -180, -168, -157, -143, -126, -108, -89, -63, -32, 1, 35, 65, 95
02818
02819
               121., 141., 152., 161., 164., 164., 161., 155., 148., 143., 137.,
02820
              133., 131., 133., 139., 150., 165., 187., 213., 248., 284., 321., 372., 449., 514., 569., 609., 642., 673., 673.
02821
02822
02823
02824
```

```
static double nua[98] = { 2120., 2125., 2130., 2135., 2140., 2145., 2150.,
2155., 2160., 2165., 2170., 2175., 2180., 2185., 2190., 2195.,
2200., 2205., 2210., 2215., 2220., 2225., 2230., 2235., 2240.,
02827
02828
             2245., 2250., 2255., 2260., 2265., 2270., 2275., 2280., 2285.,
             2290., 2295., 2300., 2305., 2310., 2315., 2320., 2325., 2330., 2335., 2340., 2345., 2350., 2355., 2360., 2365., 2370., 2375., 2380., 2385., 2390., 2395., 2400., 2405., 2410., 2415., 2420.,
02829
02830
02832
              2425., 2430., 2435., 2440., 2445., 2450., 2455., 2460., 2465.,
02833
             2470., 2475., 2480., 2485., 2490., 2495., 2500., 2505., 2510.,
02834
             2515., 2520., 2525., 2530., 2535., 2540., 2545., 2550., 2555.,
             2560., 2565., 2570., 2575., 2580., 2585., 2590., 2595., 2600., 2605.
02835
02836
02837
          double b, beta, q_n2 = 0.79, t0 = 273, tr = 296;
02838
02839
02840
          int idx;
02841
02842
           /* Check wavenumber range...
02843
          if (nu < nua[0] || nu > nua[97])
             return 0;
02844
02845
02846
          /* Interpolate B and beta... */
          idx = locate(nua, 98, nu);
b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02847
02848
02849
          beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
           /* Compute absorption coefficient... */
02851
          return 0.1 * gsl_pow_2(p / P0) * gsl_pow_2(t0 / t)
 * exp(beta * (1 / tr - 1 / t))
 * q_n2 * b * (q_n2 + (1 - q_n2) * (1.294 - 0.4545 * t / tr));
02852
02853
02854
02855 }
```

Here is the call graph for this function:



## 5.13.2.9 double ctmo2 ( double nu, double p, double t )

Compute oxygen continuum (absorption coefficient).

Definition at line 2859 of file jurassic.c.

```
02862
02863
            static double ba[90] = { 0., .061, .074, .084, .096, .12, .162, .208, .246,
02865
              .285, .314, .38, .444, .5, .571, .673, .768, .853, .966, 1.097,
02866
                1.214, 1.333, 1.466, 1.591, 1.693, 1.796, 1.922, 2.037, 2.154,
02867
                2.264, 2.375, 2.508, 2.671, 2.847, 3.066, 3.417, 3.828, 4.204,
                4.453, 4.599, 4.528, 4.284, 3.955, 3.678, 3.477, 3.346, 3.29, 3.251, 3.231, 3.226, 3.212, 3.192, 3.108, 3.033, 2.911, 2.798,
02868
02869
                2.646, 2.508, 2.322, 2.13, 1.928, 1.757, 1.588, 1.417, 1.253, 1.109, .99, .888, .791, .678, .587, .524, .464, .403, .357, .
02870
02871
02872
                .29, .267, .242, .215, .182, .16, .146, .128, .103, .087, .081,
02873
                .071, .064, 0.
02874
            };
02875
            static double betaa[90] = { 467., 467., 400., 315., 379., 368., 475., 521.,
               531., 512., 442., 444., 430., 381., 335., 324., 296., 248., 215., 193., 158., 127., 101., 71., 31., -6., -26., -47., -63., -79.,
02877
02878
               -88., -88., -87., -90., -98., -99., -109., -134., -160., -167., -164., -158., -153., -151., -156., -166., -168., -173., -170., -161., -145., -126., -108., -84., -59., -29., 4., 41., 73., 97., 123., 159., 198., 220., 242., 256., 281., 311., 334., 319., 313., 321., 323., 310., 315., 320., 335., 361., 378., 373., 338., 319.,
02879
02880
02881
02882
```

```
346., 322., 291., 290., 350., 371., 504., 504.
02885
02886
          static double nua[90] = { 1360., 1365., 1370., 1375., 1380., 1385., 1390., 1395., 1400., 1405., 1410., 1415., 1420., 1425., 1430., 1435., 1440., 1445., 1450., 1455., 1460., 1465., 1470., 1475., 1480.,
02887
02888
02889
             1485., 1490., 1495., 1500., 1505., 1510., 1515., 1520., 1525., 1530., 1535., 1540., 1545., 1550., 1555., 1560., 1565., 1570.,
02891
02892
             1575., 1580., 1585., 1590., 1595., 1600., 1605., 1610., 1615.,
02893
             1620., 1625., 1630., 1635., 1640., 1645., 1650., 1655., 1660.,
02894
             1665., 1670., 1675., 1680., 1685., 1690., 1695., 1700., 1705.,
             1710., 1715., 1720., 1725., 1730., 1735., 1740., 1745., 1750., 1755., 1760., 1765., 1770., 1775., 1780., 1785., 1790., 1795.,
02895
02896
02897
             1800., 1805.
02898
02899
          double b, beta, q_02 = 0.21, t0 = 273, tr = 296;
02900
02901
02902
          int idx;
02903
02904
           /* Check wavenumber range... */
02905
          if (nu < nua[0] || nu > nua[89])
02906
            return 0;
02907
02908
          /* Interpolate B and beta... */
02909
          idx = locate(nua, 90, nu);
b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02910
02911
          beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02912
02913
          /\star Compute absorption coefficient... \star/
          return 0.1 * gsl_pow_2(p / P0) * gsl_pow_2(t0 / t) 
 * exp(beta * (1 / tr - 1 / t)) * q_o2 * b;
02914
02915
02916 }
```

Here is the call graph for this function:



5.13.2.10 void copy\_atm ( ctl\_t \* ctl, atm\_t \* atm\_dest, atm\_t \* atm\_src, int init )

Copy and initialize atmospheric data.

Definition at line 2920 of file jurassic.c.

```
{
02925
02926
        int ig, ip, iw;
02927
02928
        size t s:
02929
02930
        /* Data size... */
02931
        s = (size_t) atm_src->np * sizeof(double);
02932
02933
        /* Copy data... */
02934
       atm_dest->np = atm_src->np;
02935
        memcpy(atm_dest->time, atm_src->time, s);
02936
        memcpy(atm_dest->z, atm_src->z, s);
02937
        memcpy(atm_dest->lon, atm_src->lon, s);
02938
        memcpy(atm_dest->lat, atm_src->lat, s);
02939
        memcpy(atm_dest->p, atm_src->p, s);
02940
        memcpy(atm_dest->t, atm_src->t, s);
02941
        for (ig = 0; ig < ctl->ng; ig++)
02942
         memcpy(atm_dest->q[ig], atm_src->q[ig], s);
02943
        for (iw = 0; iw < ctl->nw; iw++)
```

```
memcpy(atm_dest->k[iw], atm_src->k[iw], s);
02945
02946
         /* Initialize... */
02947
        if (init)
         for (ip = 0; ip < atm_dest->np; ip++) {
02948
            atm_dest->p[ip] = 0;
atm_dest->t[ip] = 0;
02949
           for (ig = 0; ig < ctl->ng; ig++)
02951
02952
               atm_dest->q[ig][ip] = 0;
            for (iw = 0; iw < ctl->nw; iw++)
atm_dest->k[iw][ip] = 0;
02953
02954
          }
02955
02956 }
```

5.13.2.11 void copy\_obs (  $ctl_t * ctl$ , obs\_t \* obs\_dest, obs\_t \* obs\_src, int init )

Copy and initialize observation data.

Definition at line 2960 of file jurassic.c.

```
02964
02966
       int id, ir;
02967
02968
        size_t s;
02969
02970
        /* Data size... */
02971
        s = (size_t) obs_src->nr * sizeof(double);
02972
02973
        /* Copy data... */
02974
        obs_dest->nr = obs_src->nr;
02975
        memcpy(obs_dest->time, obs_src->time, s);
        memcpy(obs_dest->obsz, obs_src->obsz, s);
02976
02977
        memcpy(obs_dest->obslon, obs_src->obslon, s);
02978
        memcpy(obs_dest->obslat, obs_src->obslat, s);
02979
        memcpy(obs_dest->vpz, obs_src->vpz, s);
       memcpy(obs_dest->vplon, obs_src->vplon, s);
memcpy(obs_dest->vplat, obs_src->vplat, s);
02980
02981
        memcpy(obs_dest->tpz, obs_src->tpz, s);
memcpy(obs_dest->tplon, obs_src->tplon, s);
02982
02983
02984
        memcpy(obs_dest->tplat, obs_src->tplat, s);
02985
        for (id = 0; id < ctl->nd; id++)
02986
          memcpy(obs_dest->rad[id], obs_src->rad[id], s);
02987
        for (id = 0; id < ctl->nd; id++)
02988
         memcpy(obs_dest->tau[id], obs_src->tau[id], s);
02989
02990
        /* Initialize... */
02991
        for (id = 0; id < ctl->nd; id++)
02992
02993
            for (ir = 0; ir < obs_dest->nr; ir++)
              if (gsl_finite(obs_dest->rad[id][ir])) {
02994
                obs_dest->rad[id][ir] = 0;
02995
                obs_dest->tau[id][ir] = 0;
02997
02998 }
```

5.13.2.12 int find\_emitter ( ctl t \* ctl, const char \* emitter )

Find index of an emitter.

Definition at line 3002 of file jurassic.c.

```
03004 {
03005
03006 int ig;
03007
03008 for (ig = 0; ig < ctl->ng; ig++)
03009 if (strcasecmp(ctl->emitter[ig], emitter) == 0)
03010 return ig;
03011
03012 return -1;
03013 }
```

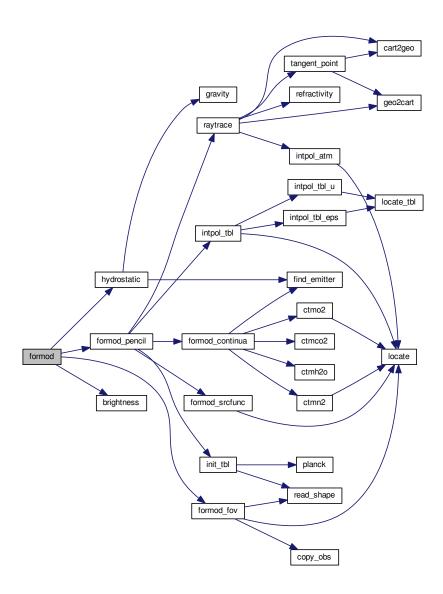
```
5.13.2.13 void formod ( ctl_t * ctl, atm_t * atm, obs_t * obs )
```

Determine ray paths and compute radiative transfer.

Definition at line 3017 of file jurassic.c.

```
03020
03021
03022
         int id, ir, *mask;
03023
03024
          /* Allocate... */
         ALLOC(mask, int,
ND * NR);
03025
03026
03027
03028
         /* Save observation mask... ∗/
         for (id = 0; id < ctl->nd; id++)
  for (ir = 0; ir < obs->nr; ir++)
    mask[id * NR + ir] = !gsl_finite(obs->rad[id][ir]);
03029
03030
03031
03032
03033
          /* Hydrostatic equilibrium... */
03034
         hydrostatic(ctl, atm);
03035
         /* Claculate pencil beams... */
for (ir = 0; ir < obs->nr; ir++)
03036
03037
03038
            formod_pencil(ctl, atm, obs, ir);
03039
         /* Apply field-of-view convolution... */
03040
03041
         formod_fov(ctl, obs);
03042
03043
          /* Convert radiance to brightness temperature... */
         if (ctl->write_bbt)
  for (id = 0; id < ctl->nd; id++)
03044
03045
              for (ir = 0; ir < obs->nr; ir++)
03046
                 obs->rad[id][ir] = brightness(obs->rad[id][ir], ctl->nu[id]);
03047
03048
03049
         /* Apply observation mask... */
for (id = 0; id < ctl->nd; id++)
  for (ir = 0; ir < obs->nr; ir++)
03050
03051
              if (mask[id * NR + ir])
  obs->rad[id][ir] = GSL_NAN;
03052
03053
03054
03055
          /* Free... */
03056
         free(mask);
03057 }
```

Here is the call graph for this function:



5.13.2.14 void formod\_continua (  $ctl_t * ctl$ ,  $los_t * los$ , int ip, double \* beta )

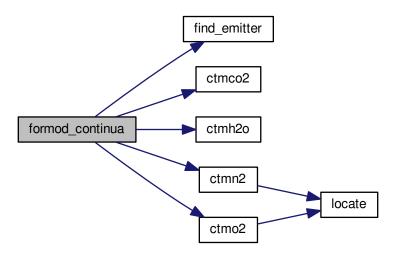
Compute absorption coefficient of continua.

Definition at line 3061 of file jurassic.c.

```
03065 {
03066
03067 static int ig_co2 = -999, ig_h2o = -999;
03068
03069 int id;
03070 /* Extinction...*/
03072 for (id = 0; id < ctl->nd; id++)
03073 beta[id] = los->k[ctl->window[id]][ip];
03074 03075 /* CO2 continuum...*/
03076 if (ctl->ctm_co2) {
03077 if (ig_co2 == -999)
```

```
ig_co2 = find_emitter(ctl, "CO2");
           if (ig_co2 >= 0)
for (id = 0; id < ctl->nd; id++)
03079
03080
                beta[id] += ctmco2(ctl->nu[id], los->p[ip], los->t[ip],
03081
03082
                                      los->u[ig_co2][ip]) / los->ds[ip];
03083
03084
03085
         /* H2O continuum... */
03086
         if (ctl->ctm_h2o) {
          if (ig_h2o == -999)
  ig_h2o = find_emitter(ctl, "H2O");
03087
03088
           if (ig_h2o >= 0)
  for (id = 0; id < ctl->nd; id++)
  beta[id] += ctmh2o(ctl->nu[id], los->p[ip], los->t[ip],
03089
03090
03091
03092
                                       los->q[ig_h2o][ip],
                                       los->u[ig_h2o][ip]) / los->ds[ip];
03093
03094
03095
03096
         /* N2 continuum... */
03097
         if (ctl->ctm_n2)
          for (id = 0; id < ct1->nd; id++)
   beta[id] += ctmn2(ct1->nu[id], los->p[ip], los->t[ip]);
03098
03099
03100
         /* 02 continuum... */
0.3101
03102
         if (ctl->ctm_o2)
03103
           for (id = 0; id < ctl->nd; id++)
03104
              beta[id] \textit{ += } ctmo2(ctl->nu[id], los->p[ip], los->t[ip]);
03105 }
```

Here is the call graph for this function:



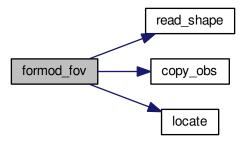
5.13.2.15 void formod\_fov (  $ctl_t * ctl$ , obs\_t \* obs )

Apply field of view convolution.

Definition at line 3109 of file jurassic.c.

```
03118
03119
        double rad[ND][NR], tau[ND][NR], wsum, z[NR], zfov;
03120
03121
        int i, id, idx, ir, ir2, nz;
03122
        /* Do not take into account FOV... */
03123
03124
        if (ctl->fov[0] == '-')
03125
03126
        /* Initialize FOV data... */
if (!init) {
03127
03128
03129
         init = 1;
03130
          read_shape(ctl->fov, dz, w, &n);
03131
03132
03133
        /* Allocate... */
03134
        ALLOC(obs2, obs_t, 1);
03135
03136
        /* Copy observation data... */
03137
        copy_obs(ctl, obs2, obs, 0);
03138
03139
        /* Loop over ray paths... */
        for (ir = 0; ir < obs->nr; ir++) {
0.3140
03141
03142
           /* Get radiance and transmittance profiles... */
03143
           for (ir2 = GSL_MAX(ir - NFOV, 0); ir2 < GSL_MIN(ir + 1 + NFOV, obs->nr);
03144
             ir2++)
if (obs->time[ir2] == obs->time[ir]) {
03145
03146
03147
              z[nz] = obs2->vpz[ir2];
for (id = 0; id < ct1->nd; id++) {
03148
                rad[id][nz] = obs2->rad[id][ir2];
tau[id][nz] = obs2->tau[id][ir2];
03149
03150
03151
03152
              nz++;
03153
          if (nz < 2)
03154
            ERRMSG("Cannot apply FOV convolution!");
03155
03156
03157
           /\star Convolute profiles with FOV... \star/
          wsum = 0;
for (id = 0; id < ctl->nd; id++) {
03158
03159
            obs->rad[id][ir] = 0;
0.3160
03161
            obs->tau[id][ir] = 0;
03162
03163
           for (i = 0; i < n; i++) {</pre>
           zfov = obs->vpz[ir] + dz[i];
idx = locate(z, nz, zfov);
03164
03165
             for (id = 0; id < ctl->nd; id++) {
03166
03167
              obs->rad[id][ir] += w[i]
               * LIN(z[idx], rad[id][idx], z[idx + 1], rad[id][idx + 1], zfov);
obs->tau[id][ir] += w[i]
03168
03169
03170
                 * LIN(z[idx], tau[id][idx], z[idx + 1], tau[id][idx + 1], zfov);
0.3171
03172
             wsum += w[i];
03173
03174
          for (id = 0; id < ctl->nd; id++) {
03175
             obs->rad[id][ir] /= wsum;
03176
             obs->tau[id][ir] /= wsum;
03177
03178
        }
03179
03180
         /* Free... */
        free(obs2);
03182 }
```

Here is the call graph for this function:



5.13.2.16 void formod\_pencil (  $ctl_t * ctl$ ,  $atm_t * atm$ ,  $obs_t * obs$ , int ir )

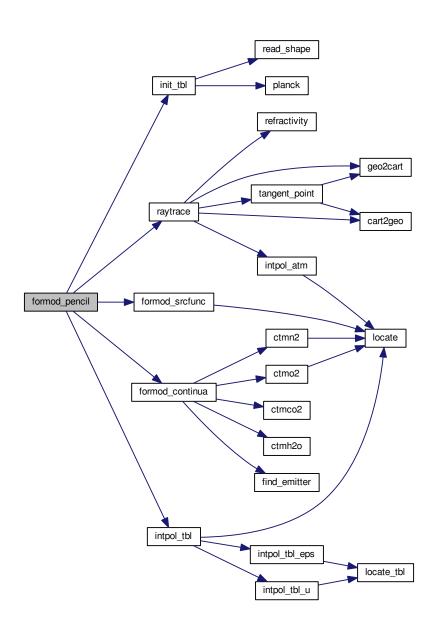
Compute radiative transfer for a pencil beam.

Definition at line 3186 of file jurassic.c.

```
03190
03191
03192
        static tbl_t *tbl;
03193
03194
        static int init = 0;
03195
03196
        los t *los:
03197
03198
        double beta_ctm[ND], eps, src_planck[ND], tau_path[NG][ND], tau_gas[ND];
03199
03200
        int id, ip;
03201
03202
        /* Initialize look-up tables... */
03203
        if (!init) {
03204
          init = 1;
03205
          ALLOC(tbl, tbl_t, 1);
03206
          init_tbl(ctl, tbl);
03207
03208
        /* Allocate... */
ALLOC(los, los_t, 1);
03209
03210
03211
        /* Initialize... */
for (id = 0; id < ctl->nd; id++) {
  obs->rad[id][ir] = 0;
03212
03213
03214
03215
          obs->tau[id][ir] = 1;
03216
03217
03218
        /* Raytracing... */
03219
        raytrace(ctl, atm, obs, los, ir);
03220
        /* Loop over LOS points... */
03221
03222
        for (ip = 0; ip < los->np; ip++) {
03223
03224
           /* Get trace gas transmittance... */
03225
           intpol_tbl(ctl, tbl, los, ip, tau_path, tau_gas);
03226
03227
           /* Get continuum absorption... */
03228
           formod_continua(ctl, los, ip, beta_ctm);
03229
03230
           /* Compute Planck function... */
03231
           formod_srcfunc(ctl, tbl, los->t[ip], src_planck);
03232
03233
          /* Loop over channels... */
for (id = 0; id < ctl->nd; id++)
03234
03235
            if (tau_gas[id] > 0) {
03236
```

```
/* Get segment emissivity... */
03238
            eps = 1 - tau_gas[id] * exp(-beta_ctm[id] * los->ds[ip]);
03239
03240
            /* Compute radiance... */
            obs->rad[id][ir] += src_planck[id] * eps * obs->tau[id][ir];
03241
03242
03243
            /\star Compute path transmittance... \star/
03244
            obs->tau[id][ir] *= (1 - eps);
03245
03246
03247
      03248
03249
03250
03251
          obs->rad[id][ir] += src_planck[id] * obs->tau[id][ir];
03252
03253
03254
      /* Free... */
03255
03256
      free(los);
03257 }
```

Here is the call graph for this function:



```
5.13.2.17 void formod_srcfunc ( ctl_t * ctl, tbl_t * tbl, double t, double * src )
```

Compute Planck source function.

Definition at line 3261 of file jurassic.c.

```
03265
                  {
03266
03267
      int id, it;
03268
03269
      /\star Determine index in temperature array... \star/
03270
      it = locate(tbl->st, TBLNS, t);
03271
03272
      /* Interpolate Planck function value... */
03273
      for (id = 0; id < ctl->nd; id++)
      03274
03275
03276 }
```

Here is the call graph for this function:



```
5.13.2.18 void geo2cart ( double z, double lon, double lat, double *x )
```

Convert geolocation to Cartesian coordinates.

Definition at line 3280 of file jurassic.c.

```
03284 {
03285
03286 double radius;
03287
03288 radius = z + RE;
03289 x[0] = radius * cos(lat / 180 * M_PI) * cos(lon / 180 * M_PI);
03290 x[1] = radius * cos(lat / 180 * M_PI) * sin(lon / 180 * M_PI);
03291 x[2] = radius * sin(lat / 180 * M_PI);
03292 }
```

5.13.2.19 double gravity ( double z, double lat )

Determine gravity of Earth.

Definition at line 3296 of file jurassic.c.

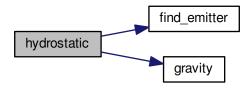
```
5.13.2.20 void hydrostatic ( ctl_t * ctl, atm_t * atm )
```

Set hydrostatic equilibrium.

Definition at line 3308 of file jurassic.c.

```
03310
03311
03312
         static int ig_h2o = -999;
03313
03314
         double dzmin = 1e99, e = 0, mean, mmair = 28.96456e-3, mmh2o =
03315
           18.0153e-3, z;
03316
03317
         int i, ip, ipref = 0, ipts = 20;
03318
         /* Check reference height... */
03320
         if (ctl->hydz < 0)
            return;
03321
03322
03323
         /* Determine emitter index of H2O... */
03324
         if (ig_h2o == -999)
03325
           ig_h2o = find_emitter(ctl, "H2O");
03326
03327
          /\star Find air parcel next to reference height... \star/
         for (ip = 0; ip < atm->np; ip++)
  if (fabs(atm->z[ip] - ctl->hydz) < dzmin) {
    dzmin = fabs(atm->z[ip] - ctl->hydz);
03328
03329
03330
              ipref = ip;
03331
03332
03333
         /* Upper part of profile... */
for (ip = ipref + 1; ip < atm->np; ip++) {
03334
03335
03336
           mean = 0;
            for (i = 0; i < ipts; i++) {</pre>
03337
03338
              z = LIN(0.0, atm \rightarrow z[ip - 1], ipts - 1.0, atm \rightarrow z[ip], (double) i);
03339
              if (ig_h2o >= 0)
03340
                e = LIN(0.0, atm->q[ig_h2o][ip - 1],
03341
              ipts - 1.0, atm->q[ig_h2o][ip], (double) i); mean += (e * mmh2o + (1 - e) * mmair)
03342
03343
                 * gravity(z, atm->lat[ipref]) / GSL_CONST_MKSA_MOLAR_GAS
03344
                 / LIN(0.0, atm->t[ip - 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03345
03346
           /* Compute p(z,T)... */
03347
03348
           atm->p[ip] =
03349
              \exp(\log(\arctan - p[ip - 1]) - mean * 1000 * (atm - z[ip] - atm - z[ip - 1]));
03350
03351
03352
         /\star Lower part of profile... \star/
03353
         for (ip = ipref - 1; ip >= 0; ip--) {
03354
           mean = 0;
            for (i = 0; i < ipts; i++) {</pre>
03355
03356
              z = LIN(0.0, atm \rightarrow z[ip + 1], ipts - 1.0, atm \rightarrow z[ip], (double) i);
03357
              if (ig_h2o >= 0)
03358
                e = LIN(0.0, atm->q[ig_h2o][ip + 1],
              ipts - 1.0, atm->q[ig_h2o][ip], (double) i);
mean += (e * mmh2o + (1 - e) * mmair)
  * gravity(z, atm->lat[ipref]) / GSL_CONST_MKSA_MOLAR_GAS
03359
03360
03361
03362
                 / LIN(0.0, atm->t[ip + 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03363
03364
03365
            /* Compute p(z,T) \dots */
03366
           atm->p[ip]
03367
              \exp(\log(\text{atm->p[ip + 1]}) - \text{mean} * 1000 * (\text{atm->z[ip] - atm->z[ip + 1]}));
03368
03369 }
```

Here is the call graph for this function:



```
5.13.2.21 void idx2name (ctl t * ctl, int idx, char * quantity)
```

Determine name of state vector quantity for given index.

Definition at line 3373 of file jurassic.c.

```
03376
03378
          int ig, iw;
03379
03380
          if (idx == IDXP)
             sprintf(quantity, "PRESSURE");
03381
03382
          if (idx == IDXT)
03383
03384
             sprintf(quantity, "TEMPERATURE");
03385
          for (ig = 0; ig < ctl->ng; ig++)
  if (idx == IDXQ(ig))
    sprintf(quantity, "%s", ctl->emitter[ig]);
03386
03387
03388
03389
          for (iw = 0; iw < ctl->nw; iw++)
  if (idx == IDXK(iw))
    sprintf(quantity, "EXTINCT_WINDOW%d", iw);
03390
03391
03392
03393 }
```

```
5.13.2.22 void init_tbl ( ctl_t * ctl, tbl_t * tbl )
```

Initialize look-up tables.

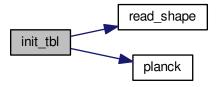
Definition at line 3397 of file jurassic.c.

```
03399
                         {
03400
03401
         FILE *in;
03402
         char filename[LEN], line[LEN];
03403
03404
         double eps, eps_old, press, press_old, temp, temp_old, u, u_old,
   f[NSHAPE], fsum, nu[NSHAPE];
03405
03406
03407
03408
         int i, id, ig, ip, it, n;
03409
         /* Loop over trace gases and channels... */
03410
03411 for (ig = 0; ig < ctl->ng; ig++)
03412 #pragma omp parallel for default (none) shared(ctl,tbl,ig) private(in,filename,line,eps,eps_old,press,
      press_old,temp,temp_old,u,u_old,id,ip,it)
03413
           for (id = 0; id < ctl->nd; id++) {
03414
03415
              /* Initialize... */
             tbl->np[ig][id] = -1;
eps_old = -999;
03416
03417
```

```
03418
             press_old = -999;
              temp\_old = -999;
03419
03420
             u_old = -999;
03421
             03422
03423
03424
03425
              if (!(in = fopen(filename, "r"))) {
03426
              printf("Missing emissivity table: %s\n", filename);
03427
                continue;
03428
             printf("Read emissivity table: %s\n", filename);
03429
03430
03431
              /* Read data... */
03432
              while (fgets(line, LEN, in)) {
03433
                /* Parse line... */ if (sscanf(line, "%lg %lg %lg %lg", &press, &temp, &u, &eps) != 4)
03434
03435
03436
                  continue;
03437
03438
                /* Determine pressure index... */
                if (press != press_old) {
  press_old = press;
  if ((++tbl->np[ig][id]) >= TBLNP)
03439
03440
03441
                  ERRMSG("Too many pressure levels!");
tbl->nt[ig][id][tbl->np[ig][id]] = -1;
03442
03443
03444
03445
03446
                /\star Determine temperature index... \star/
                if (temp != temp_old) {
  temp_old = temp;
03447
03448
03449
                      ((++tbl->nt[ig][id][tbl->np[ig][id]]) >= TBLNT)
03450
                    ERRMSG("Too many temperatures!");
03451
                  tbl->nu[ig][id][tbl->np[ig][id]]
03452
                    [tbl->nt[ig][id][tbl->np[ig][id]]] = -1;
03453
03454
03455
                /* Determine column density index... */
03456
                if ((eps > eps_old && u > u_old) || tbl->nu[ig][id][tbl->np[ig][id]]
03457
                    [tbl->nt[ig][id][tbl->np[ig][id]]] < 0) {
03458
                  eps_old = eps;
                  u\_old = u;
03459
                  if ((++tbl->nu[ig][id][tbl->np[ig][id]]
03460
                        [tbl->nt[ig][id][tbl->np[ig][id]]]) >= TBLNU) {
03461
                    tbl->nu[ig][id][tbl->np[ig][id]]
03462
03463
                       [tbl->nt[ig][id][tbl->np[ig][id]]]--;
03464
                    continue;
03465
                  }
               }
03466
03467
03468
                /* Store data... */
03469
                tbl->p[ig][id][tbl->np[ig][id]] = press;
03470
                \label{tbl->t[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]} tbl->t[ig][id][tbl->np[ig][id]]]
03471
                  = temp;
                tbl->u[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
[tbl->nu[ig][id][tbl->np[ig][id]]
[tbl->nt[ig][id][tbl->np[ig][id]]] = (float) u;
03472
03473
03474
03475
                tbl->eps[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
03476
                  [tbl->nu[ig][id][tbl->np[ig][id]]
03477
                    [tbl->nt[ig][id][tbl->np[ig][id]]] = (float) eps;
03478
03479
03480
              /* Increment counters... */
03481
              tbl->np[ig][id]++;
03482
              for (ip = 0; ip < tbl->np[ig][id]; ip++) {
03483
                tbl->nt[ig][id][ip]++;
                for (it = 0; it < tbl->nt[ig][id][ip]; it++)
  tbl->nu[ig][id][ip][it]++;
03484
03485
03486
03487
03488
              /* Close file... */
03489
             fclose(in);
03490
          }
03491
03492
        /* Write info... */
        printf("Initialize source function table...\n");
03493
03494
03495
        /* Loop over channels... */
03496 #pragma omp parallel for default(none) shared(ctl,tbl,ig) private(filename,it,i,n,f,fsum,nu) 03497 for (id = 0; id < ctl->nd; id++) {
03498
03499
            /* Read filter function... */
03500
           sprintf(filename, "%s_%.4f.filt", ctl->tblbase, ctl->nu[id]);
03501
           read_shape(filename, nu, f, &n);
03502
           /* Compute source function table... */
for (it = 0; it < TBLNS; it++) {</pre>
03503
03504
```

```
/* Set temperature... */ tbl->st[it] = LIN(0.0, TMIN, TBLNS - 1.0, TMAX, (double) it);
03506
03507
03508
03509
             /* Integrate Planck function... */
03510
             fsum = 0;
03511
             tbl->sr[id][it] = 0;
03512
             for (i = 0; i < n; i++) {</pre>
03513
              fsum += f[i];
               tbl->sr[id][it] += f[i] * planck(tbl->st[it], nu[i]);
03514
03515
03516
             tbl->sr[id][it] /= fsum;
03517
03518
03519 }
```

Here is the call graph for this function:



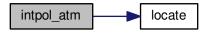
5.13.2.23 void intpol\_atm ( ctl t \* ctl, atm\_t \* atm, double z, double \* p, double \* t, double \* q, double \* k)

Interpolate atmospheric data.

Definition at line 3523 of file jurassic.c.

```
03530
                          {
03531
03532
          int ig, ip, iw;
03533
03534
          /* Get array index... */
03535
          ip = locate(atm->z, atm->np, z);
03536
          /* Interpolate... */
          *p = EXP(atm->z[ip], atm->p[ip], atm->z[ip + 1], atm->p[ip + 1], z);
*t = LIN(atm->z[ip], atm->t[ip], atm->z[ip + 1], atm->t[ip + 1], z);
03538
03539
          for (ig = 0; ig < ctl->ng; ig++)
  q[ig] =
03540
03541
          LIN(atm->z[ip], atm->q[ig][ip], atm->z[ip + 1], atm->q[ig][ip + 1], z); for (iw = 0; iw < ctl->nw; iw++)
03542
03543
03544
03545
                \label{eq:linear} LIN\,(atm->z\,[ip],\ atm->k\,[iw]\,[ip],\ atm->z\,[ip+1],\ atm->k\,[iw]\,[ip+1],\ z)\,;
03546 }
```

Here is the call graph for this function:



5.13.2.24 void intpol\_tbl ( ctl\_t \* ctl, tbl\_t \* tbl, los\_t \* los, int ip, double tau\_path[NG][ND], double tau\_seg[ND] )

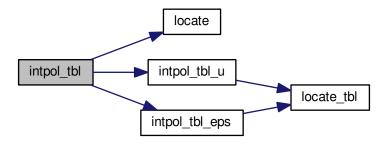
Get transmittance from look-up tables.

Definition at line 3550 of file jurassic.c.

```
03556
03557
03558
        double eps, eps00, eps01, eps10, eps11, u;
03559
03560
        int id, ig, ipr, it0, it1;
03561
03562
        /* Initialize... */
03563
        if (ip <= 0)</pre>
03564
         for (ig = 0; ig < ctl->ng; ig++)
03565
            for (id = 0; id < ctl->nd; id++)
03566
              tau_path[ig][id] = 1;
03567
03568
        /* Loop over channels... */
03569
        for (id = 0; id < ctl->nd; id++) {
03570
           /* Initialize... */
03571
03572
          tau_seg[id] = 1;
03573
03574
           /* Loop over emitters.... */
03575
          for (ig = 0; ig < ctl->ng; ig++) {
03576
03577
             /\star Check size of table (pressure)... \star/
03578
             if (tbl->np[ig][id] < 2)</pre>
03579
              eps = 0;
03580
03581
             /* Check transmittance... */
             else if (tau_path[ig][id] < 1e-9)</pre>
03583
              eps = 1;
03584
03585
             /* Interpolate... */
03586
             else {
03587
03588
               /* Determine pressure and temperature indices... */
               ipr = locate(tbl->p[ig][id], tbl->np[ig][id], los->p[ip]);
it0 = locate(tbl->t[ig][id][ipr], tbl->nt[ig][id][ipr], los->t[ip]);
03589
03590
               it1 =
03591
03592
                 locate(tbl->t[ig][id][ipr + 1], tbl->nt[ig][id][ipr + 1],
03593
                         los->t[ip]);
03594
03595
               /\star Check size of table (temperature and column density)... \star/
03596
               if (tbl->nt[ig][id][ipr] < 2 || tbl->nt[ig][id][ipr + 1] < 2</pre>
                   || tbl->nu[ig][id][ipr][it0] < 2
|| tbl->nu[ig][id][ipr][it0 + 1] < 2
03597
03598
03599
                   || tbl->nu[ig][id][ipr + 1][it1] < 2
|| tbl->nu[ig][id][ipr + 1][it1 + 1] < 2)
03600
03601
                 eps = 0;
03602
03603
               else {
03604
                 /* Get emissivities of extended path... */
u = intpol_tbl_u(tbl, ig, id, ipr, it0, 1 - tau_path[ig][id]);
eps00 = intpol_tbl_eps(tbl, ig, id, ipr, it0, u + los->u[ig][ip]);
03605
03606
03607
03608
03609
                 u = intpol_tbl_u(tbl, ig, id, ipr, it0 + 1, 1 - tau_path[ig][id]);
03610
                 eps01 =
                   intpol_tbl_eps(tbl, ig, id, ipr, it0 + 1, u + los->u[ig][ip]);
03611
03612
03613
                 u = intpol_tbl_u(tbl, ig, id, ipr + 1, it1, 1 - tau_path[ig][id]);
03614
                 eps10 =
03615
                   intpol_tbl_eps(tbl, ig, id, ipr + 1, it1, u + los->u[ig][ip]);
03616
03617
03618
                   intpol tbl u(tbl, ig, id, ipr + 1, it1 + 1, 1 - tau path[ig][id]);
03619
                 eps11
                   intpol_tbl_eps(tbl, ig, id, ipr + 1, it1 + 1, u + los->
      u[ig][ip]);
03621
03622
                 /* Interpolate with respect to temperature... */
                 03623
03624
                 eps11 = LIN(tbl->t[ig][id][ipr + 1][it1], eps10,
03625
03626
                              tbl->t[ig][id][ipr + 1][it1 + 1], eps11, los->t[ip]);
03627
03628
                 /* Interpolate with respect to pressure... */
                 03629
03630
03631
```

```
/* Check emssivity range... */
03633
                  eps00 = GSL_MAX(GSL_MIN(eps00, 1), 0);
03634
                  /* Determine segment emissivity... */
eps = 1 - (1 - eps00) / tau_path[ig][id];
03635
03636
                }
03637
03638
03639
03640
              /\star Get transmittance of extended path... \star/
03641
              tau_path[ig][id] *= (1 - eps);
03642
03643
              /* Get segment transmittance... */
03644
             tau_seg[id] *= (1 - eps);
03645
03646
03647 }
```

Here is the call graph for this function:



5.13.2.25 double intpol\_tbl\_eps ( tbl\_t \* tbl, int ig, int id, int ip, int it, double u )

Interpolate emissivity from look-up tables.

Definition at line 3651 of file jurassic.c.

```
03657
                    {
03658
03659
         int idx;
03660
03661
         /* Lower boundary... */
03662
         if (u < tbl->u[ig][id][ip][it][0])
          return LIN(0, 0, tbl->u[ig][id][ip][it][0], tbl->eps[ig][id][ip][it][0],
03663
03664
                       u);
03665
03666
         /* Upper boundary... */
         else if (u > tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
03667
           return LIN(tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03668
                       tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03669
03670
                       1e30, 1, u);
03671
03672
         /* Interpolation... */
03673
         else {
03674
           /* Get index... */  idx = locate\_tbl(tbl->u[ig][id][ip][it], \ tbl->nu[ig][id][ip][it], \ u); 
03675
03676
03677
03678
           /* Interpolate... */
03679
             LIN(tbl->u[ig][id][ip][it][idx], tbl->eps[ig][id][ip][it][idx], tbl->u[ig][id][ip][it][idx + 1], tbl->eps[ig][id][ip][it][idx + 1],
03680
03681
03682
                  u);
03683
         }
03684 }
```

Here is the call graph for this function:



```
5.13.2.26 double intpol_tbl_u ( tbl_t * tbl, int ig, int id, int ip, int it, double eps )
```

Interpolate column density from look-up tables.

Definition at line 3688 of file jurassic.c.

```
03694
03696
      int idx;
03697
      /* Lower boundary... */
if (eps < tbl->eps[ig][id][ip][it][0])
  return LIN(0, 0, tbl->eps[ig][id][ip][it][0], tbl->u[ig][id][ip][it][0],
03698
03699
03700
03701
                  eps);
03702
      03703
03704
03705
03706
                  1, 1e30, eps);
03708
03709
      /* Interpolation... */
03710
      else {
03711
03712
        /* Get index... */
03713
        idx = locate_tbl(tbl->eps[ig][id][ip][it], tbl->nu[ig][id][ip][it], eps);
03714
03715
03716
        return
03717
         03718
03719
             eps);
03720
03721 }
```

Here is the call graph for this function:



5.13.2.27 void jsec2time ( double jsec, int \* year, int \* mon, int \* day, int \* hour, int \* min, int \* sec, double \* remain )

Convert seconds to date.

Definition at line 3725 of file jurassic.c.

```
03733
03734
03735
        struct tm t0, *t1;
03737
        time_t jsec0;
03738
03739
        t0.tm_year = 100;
        t0.tm_mon = 0;
03740
03741
        t0.tm_mday = 1;
03742
        t0.tm\_hour = 0;
03743
        t0.tm_min = 0;
        t0.tm_sec = 0;
03744
03745
03746
        jsec0 = (time_t) jsec + timegm(&t0);
03747
        t1 = gmtime(&jsec0);
03748
03749
        *year = t1->tm_year + 1900;
03750
        *mon = t1->tm_mon + 1;
        *day = t1->tm_mday;
03751
03752
        *hour = t1->tm_hour;
03753
        *min = t1->tm_min;
        *sec = t1->tm_sec;
*remain = jsec - floor(jsec);
03754
03755
03756 }
```

5.13.2.28 void kernel (  $ctl_t * ctl$ ,  $atm_t * atm$ ,  $obs_t * obs$ ,  $gsl_matrix * k$  )

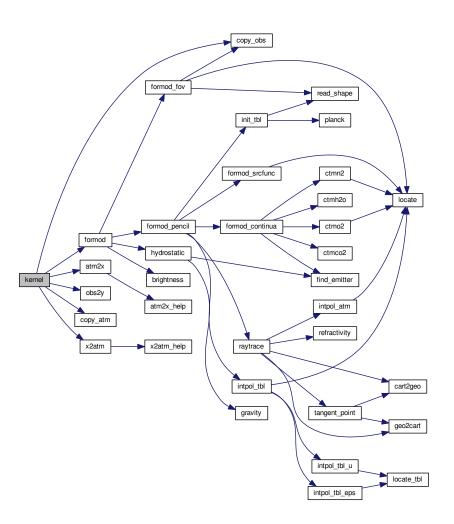
Compute Jacobians.

Definition at line 3760 of file jurassic.c.

```
03764
                                                                              {
03765
03766
                         atm_t *atm1;
03767
                        obs_t *obs1;
03768
03769
                        gsl_vector *x0, *x1, *yy0, *yy1;
03770
03771
                         int *iqa, j;
03772
03773
                        double h;
03774
03775
                        size_t i, n, m;
03776
03777
                        /* Get sizes... */
                        m = k->size1;
n = k->size2;
03778
03779
03780
03781
                         /* Allocate... */
03782
                        x0 = gsl_vector_alloc(n);
                         yy0 = gsl_vector_alloc(m);
03784
                         ALLOC(iqa, int,
03785
                                             N);
03786
03787
                         /* Compute radiance for undisturbed atmospheric data... */
03788
                         formod(ctl, atm, obs);
03789
03790
                         /* Compose vectors... */
03791
                         atm2x(ctl, atm, x0, iqa, NULL);
03792
                         obs2y(ctl, obs, yy0, NULL, NULL);
03793
03794
                        /* Initialize kernel matrix... */
03795
                        gsl_matrix_set_zero(k);
03796
03797
                          /\star Loop over state vector elements... \star/
03798 \text{ \#pragma omp parallel for default(none) shared(ctl,atm,obs,k,x0,yy0,n,m,iqa) private(i, j, h, x1, yy1, atm1, atm1, atm2, a
                      obs1)
03799
                         for (j = 0; j < (int) n; j++) {
03800
03801
                               /* Allocate... */
```

```
x1 = gsl_vector_alloc(n);
03803
           yy1 = gsl_vector_alloc(m);
03804
           ALLOC(atm1, atm_t, 1);
           ALLOC(obs1, obs_t, 1);
03805
03806
03807
           /* Set perturbation size... */
03808
           if (iqa[j] == IDXP)
03809
             h = GSL_MAX(fabs(0.01 * gsl_vector_get(x0, (size_t) j)), 1e-7);
03810
           else if (iqa[j] == IDXT)
             h = 1;
03811
           else if (iqa[j] >= IDXQ(0) && iqa[j] < IDXQ(ctl->ng))
03812
           \label{eq:heat_max} \begin{array}{ll} h = GSL\_MAX(fabs(0.01 * gsl\_vector\_get(x0, (size\_t) j)), \ 1e-15); \\ else \ if \ (iqa[j] >= IDXK(0) \ \&\& \ iqa[j] < IDXK(ctl->nw)) \end{array}
03813
03814
03815
03816
           else
             ERRMSG("Cannot set perturbation size!");
03817
03818
03819
           /* Disturb state vector element... */
03820
           gsl_vector_memcpy(x1, x0);
03821
           gsl_vector_set(x1, (size_t) j, gsl_vector_get(x1, (size_t) j) + h);
           copy_atm(ctl, atm1, atm, 0);
copy_obs(ctl, obs1, obs, 0);
03822
03823
03824
           x2atm(ctl, x1, atm1);
03825
03826
            /* Compute radiance for disturbed atmospheric data... */
           formod(ctl, atml, obsl);
03828
03829
            /\star Compose measurement vector for disturbed radiance data... \star/
03830
           obs2y(ctl, obs1, yy1, NULL, NULL);
03831
03832
           /* Compute derivatives... */
03833
           for (i = 0; i < m; i++)
03834
             gsl_matrix_set(k, i, (size_t) j,
03835
                               (gsl_vector_get(yy1, i) - gsl_vector_get(yy0, i)) / h);
03836
           /* Free... */
03837
           gsl_vector_free(x1);
gsl_vector_free(yy1);
03838
03839
03840
           free(atm1);
03841
           free (obs1);
03842
03843
        /* Free... */
gsl_vector_free(x0);
03844
03845
03846
         gsl_vector_free(yy0);
03847
         free(iqa);
03848 }
```

Here is the call graph for this function:



5.13.2.29 int locate ( double \*xx, int n, double x )

Find array index.

Definition at line 3852 of file jurassic.c.

```
{
03856
          int i, ilo, ihi;
03857
03858
03859
          ilo = 0;
         ihi = n - 1;
i = (ihi + ilo) >> 1;
03860
03861
03862
          if (xx[i] < xx[i + 1])
while (ihi > ilo + 1) {
   i = (ihi + ilo) >> 1;
   if (xx[i] > x)
      ihi = i;
03863
03864
03865
03866
03867
               else
03868
03869
                 ilo = i;
         03870
03871
03872
03873
03874
                 ihi = i;
```

```
03875 else
03876 ilo = i;
03877 }
03878 03879 return ilo;
03880 }
```

5.13.2.30 int locate\_tbl (float \* xx, int n, double x)

Find array index in float array.

Definition at line 3884 of file jurassic.c.

```
03887
                  {
03888
03889
       int i, ilo, ihi;
03890
03891
       ilo = 0;
03892
       ihi = n - 1;
03893
       i = (ihi + ilo) >> 1;
03894
       while (ihi > ilo + 1) {
03895
        i = (ihi + ilo) >> 1;
if (xx[i] > x)
03896
03897
03898
            ihi = i;
03899
         else
03900
            ilo = i;
       }
03901
03902
03903
       return ilo;
03904 }
```

5.13.2.31 size\_t obs2y ( ctl\_t \* ctl, obs\_t \* obs, gsl\_vector \* y, int \* ida, int \* ira )

Compose measurement vector.

Definition at line 3908 of file jurassic.c.

```
03913
                    {
03914
03915
        int id, ir;
03916
03917
        size_t m = 0;
03918
03919
        /* Determine measurement vector... */
03920
        for (ir = 0; ir < obs->nr; ir++)
        for (id = 0; id < ctl->nd; id++)
03921
03922
            if (gsl_finite(obs->rad[id][ir])) {
             if (y != NULL)
  gsl_vector_set(y, m, obs->rad[id][ir]);
if (ida != NULL)
03923
03924
03925
              ida[m] = id;
if (ira != NULL)
03926
03927
03928
                 ira[m] = ir;
03929
              m++;
03930
03931
03932
       return m;
03933 }
```

5.13.2.32 double planck (double t, double nu)

Compute Planck function.

Definition at line 3937 of file jurassic.c.

```
5.13.2.33 void raytrace ( ctl_t * ctl, atm_t * atm, obs_t * obs, los_t * los, int ir )
```

Do ray-tracing to determine LOS.

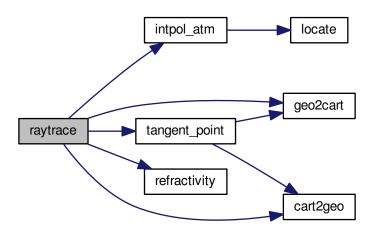
Definition at line 3946 of file jurassic.c.

```
03951
                 {
03952
03953
        double cosa, d, dmax, dmin = 0, ds, ex0[3], ex1[3], frac, h = 0.02, k[NW],
03954
         lat, lon, n, naux, ng[3], norm, p, q[NG], t, x[3], xh[3],
03955
          xobs[3], xvp[3], z = 1e99, zmax, zmin, zrefrac = 60;
03956
03957
        int i, ig, ip, iw, stop = 0;
03959
        /* Initialize... */
03960
        los->np = 0;
        los->tsurf = -999;
obs->tpz[ir] = obs->vpz[ir];
03961
03962
03963
        obs->tplon[ir] = obs->vplon[ir];
03964
        obs->tplat[ir] = obs->vplat[ir];
03965
03966
        /* Get altitude range of atmospheric data... */
03967
        gsl_stats_minmax(&zmin, &zmax, atm->z, 1, (size_t) atm->np);
03968
03969
        /* Check observer altitude... */
03970
           (obs->obsz[ir] < zmin)
03971
          ERRMSG("Observer below surface!");
03972
03973
        /* Check view point altitude... */
03974
        if (obs->vpz[ir] > zmax)
03975
         return;
03976
03977
        /* Determine Cartesian coordinates for observer and view point... */
03978
        geo2cart(obs->obsz[ir], obs->obslon[ir], obs->obslat[ir], xobs);
03979
        geo2cart(obs->vpz[ir], obs->vplon[ir], obs->vplat[ir], xvp);
03980
        /* Determine initial tangent vector... */
03981
03982
        for (i = 0; i < 3; i++)
03983
          ex0[i] = xvp[i] - xobs[i];
        norm = NORM(ex0);
for (i = 0; i < 3; i++)
03984
03985
03986
          ex0[i] /= norm;
03987
03988
        /* Observer within atmosphere... */
        for (i = 0; i < 3; i++)
03989
03990
          x[i] = xobs[i];
03991
03992
        /* Observer above atmosphere (search entry point)... */
03993
        if (obs->obsz[ir] > zmax) {
03994
          dmax = norm;
03995
          while (fabs(dmin - dmax) > 0.001) {
03996
            d = (dmax + dmin) / 2;
            for (i = 0; i < 3; i++)
x[i] = xobs[i] + d * ex0[i];
03997
03998
03999
            cart2geo(x, &z, &lon, &lat);
            if (z \le zmax && z > zmax - 0.001)
04000
04001
              break;
04002
             if (z < zmax - 0.0005)
04003
              dmax = d;
04004
            else
              dmin = d;
04005
04006
          }
04007
        }
04008
04009
        /* Ray-tracing... */
04010
        while (1) {
04011
04012
          /* Set step length... */
04013
          ds = ctl->ravds;
          if (ctl->raydz > 0) {
04014
04015
            norm = NORM(x);
            for (i = 0; i < 3; i++)
   xh[i] = x[i] / norm;</pre>
04016
04017
             cosa = fabs(DOTP(ex0, xh));
04018
04019
            if (cosa != 0)
04020
              ds = GSL_MIN(ctl->rayds, ctl->raydz / cosa);
04021
04022
04023
          /* Determine geolocation... */
04024
          cart2geo(x, &z, &lon, &lat);
04025
04026
          /* Check if LOS hits the ground or has left atmosphere... */
04027
          if (z < zmin || z > zmax) {
```

```
04028
              stop = (z < zmin ? 2 : 1);
04029
               ((z <
04030
04031
                  zmin ? zmin : zmax) - los-z[los-np - 1]) / (z - los-z[los-np - 1])
04032
                                                                                      11);
              geo2cart(los->z[los->np - 1], los->lon[los->np - 1],
04033
              los->lat[los->np - 1], xh);
for (i = 0; i < 3; i++)
04034
04035
04036
               x[i] = xh[i] + frac * (x[i] - xh[i]);
04037
              cart2geo(x, &z, &lon, &lat);
              los->ds[los->np - 1] = ds * frac;
04038
04039
              ds = 0:
04040
04041
04042
            /* Interpolate atmospheric data... */
04043
           intpol_atm(ctl, atm, z, &p, &t, q, k);
04044
04045
            /* Save data... */
           los -> lon[los -> np] = lon;
04046
04047
           los->lat[los->np] = lat;
04048
           los \rightarrow z[los \rightarrow np] = z;
04049
           los \rightarrow p[los \rightarrow np] = p;
           los->t[los->np] = t;
04050
           for (ig = 0; ig < ctl->ng; ig++)
los->q[ig][los->np] = q[ig];
04051
04052
04053
           for (iw = 0; iw < ctl->nw; iw++)
04054
             los \rightarrow k[iw][los \rightarrow np] = k[iw];
04055
           los->ds[los->np] = ds;
04056
04057
           /* Increment and check number of LOS points... */
04058
           if ((++los->np) > NLOS)
04059
              ERRMSG("Too many LOS points!");
04060
04061
            /* Check stop flag... */
04062
           if (stop) {
             los->tsurf = (stop == 2 ? t : -999);
04063
04064
             break;
04065
04066
04067
            /* Determine refractivity... */
04068
           if (ctl->refrac && z <= zrefrac)</pre>
04069
             n = 1 + refractivity(p, t);
04070
           else
04071
             n = 1;
04072
04073
           /\star Construct new tangent vector (first term)... \star/
04074
           for (i = 0; i < 3; i++)
              ex1[i] = ex0[i] * n;
04075
04076
04077
           /* Compute gradient of refractivity... */
           if (ctl->refrac && z <= zrefrac) {
             for (i = 0; i < 3; i++)
xh[i] = x[i] + 0.5 * ds * ex0[i];
04079
04080
              cart2geo(xh, &z, &lon, &lat);
intpol_atm(ctl, atm, z, &p, &t, q, k);
n = refractivity(p, t);
for (i = 0; i < 3; i++) {</pre>
04081
04082
04083
04084
                xh[i] += h;
04085
04086
                cart2geo(xh, &z, &lon, &lat);
                intpol_atm(ctl, atm, z, &p, &t, q, k);
naux = refractivity(p, t);
04087
04088
                ng[i] = (naux - n) / h;
04089
04090
                xh[i] -= h;
04091
04092
           } else
04093
              for (i = 0; i < 3; i++)
               ng[i] = 0;
04094
04095
04096
           /* Construct new tangent vector (second term)... */
           for (i = 0; i < 3; i++)
04097
04098
             ex1[i] += ds * ng[i];
04099
04100
           /\star Normalize new tangent vector... \star/
04101
           norm = NORM(ex1);
           for (i = 0; i < 3; i++)
ex1[i] /= norm;
04102
04103
04104
04105
            /* Determine next point of LOS... */
           for (i = 0; i < 3; i++)
x[i] += 0.5 * ds * (ex0[i] + ex1[i]);</pre>
04106
04107
04108
04109
           /* Copy tangent vector... */
for (i = 0; i < 3; i++)</pre>
04110
04111
              ex0[i] = ex1[i];
04112
04113
04114
         /* Get tangent point (to be done before changing segment lengths!)... */
```

```
04115
       tangent_point(los, &obs->tpz[ir], &obs->tplon[ir], &obs->
04116
04117
        /\star Change segment lengths according to trapezoid rule... \star/
       for (ip = los->np - 1; ip >= 1; ip--)
los->ds[ip] = 0.5 * (los->ds[ip - 1] + los->ds[ip]);
04118
04119
04120
       los -> ds[0] *= 0.5;
04121
04122
        /* Compute column density... */
       04123
04124
04125
04126
04127 }
```

Here is the call graph for this function:



5.13.2.34 void read\_atm ( const char \* dirname, const char \* filename, ctl t \* ctl, atm t \* atm )

Read atmospheric data.

Definition at line 4131 of file jurassic.c.

```
04135
04136
04137
        FILE *in;
04138
04139
        char file[LEN], line[LEN], *tok;
04140
04141
       int ig, iw;
04142
04143
       /* Init... */
04144
       atm->np = 0;
04145
04146
        /* Set filename... */
        if (dirname != NULL)
04147
04148
         sprintf(file, "%s/%s", dirname, filename);
04149
04150
         sprintf(file, "%s", filename);
04151
04152
        /* Write info... */
       printf("Read atmospheric data: %s\n", file);
04153
04154
04155
        /* Open file... */
04156
       if (!(in = fopen(file, "r")))
04157
         ERRMSG("Cannot open file!");
```

```
04158
04159
                /* Read line... */
04160
                while (fgets(line, LEN, in)) {
04161
                  /* Read data... */
TOK(line, tok, "%lg", atm->time[atm->np]);
TOK(NULL, tok, "%lg", atm->z[atm->np]);
TOK(NULL, tok, "%lg", atm->z[atm->np]);
TOK(NULL, tok, "%lg", atm->lat[atm->np]);
TOK(NULL, tok, "%lg", atm->p[atm->np]);
TOK(NULL, tok, "%lg", atm->t[atm->np]);
TOK(NULL, tok, "%lg", atm->t[atm->np]);
for (ig = 0; ig < ctl->ng; ig++)
TOK(NULL, tok, "%lg", atm->q[ig][atm->np]);
for (iw = 0; iw < ctl->nw; iw++)
TOK(NULL, tok, "%lg", atm->k[iw][atm->np]);
04162
04163
04164
04165
04166
04167
04168
04169
04170
04171
04172
04173
                 /* Increment data point counter... */
if ((++atm->np) > NP)
04174
04175
                       ERRMSG("Too many data points!");
04177
04178
04179
               /* Close file... */
04180
               fclose(in);
04181
04182
                /* Check number of points... */
04183
                if (atm->np < 1)
04184
                    ERRMSG("Could not read any data!");
04185 }
```

5.13.2.35 void read\_ctl ( int argc, char \* argv[], ctl\_t \* ctl )

Read forward model control parameters.

Definition at line 4189 of file jurassic.c.

```
04192
04193
04194
         int id, ig, iw;
04195
04196
         /* Write info... */
         printf("\nJuelich Rapid Spectral Simulation Code (JURASSIC)\n"
04197
                   "(executable: %s | compiled: %s, %s)\n\n",
04198
04199
                   argv[0], __DATE__, __TIME__);
04200
04201
         /* Emitters... */
         ctl->ng = (int) scan_ctl(argc, argv, "NG", -1, "0", NULL); if (ctl->ng < 0 || ctl->ng > NG)
04202
04203
            ERRMSG("Set 0 <= NG <= MAX!");
04204
04205
          for (ig = 0; ig < ctl->ng; ig++)
04206
            scan_ctl(argc, argv, "EMITTER", ig, "", ctl->emitter[ig]);
04207
         /* Radiance channels... */
ctl->nd = (int) scan_ctl(argc, argv, "ND", -1, "0", NULL);
if (ctl->nd < 0 || ctl->nd > ND)
04208
04209
04210
04211
            ERRMSG("Set 0 <= ND <= MAX!");</pre>
04212
          for (id = 0; id < ctl->nd; id++)
04213
            ctl->nu[id] = scan_ctl(argc, argv, "NU", id, "", NULL);
04214
04215
         /* Spectral windows... */
04216
         ctl->nw = (int) scan_ctl(argc, argv, "NW", -1, "1", NULL);
04217
         if (ctl->nw < 0 || ctl->nw > NW)
04218
            ERRMSG("Set 0 <= NW <= MAX!");</pre>
04219
          for (id = 0; id < ctl->nd; id++)
04220
            ctl->window[id] = (int) scan_ctl(argc, argv, "WINDOW", id, "0", NULL);
04221
         /* Emissivity look-up tables... */
scan_ctl(argc, argv, "TBLBASE", -1, "-", ctl->tblbase);
04222
04223
04224
04225
          /* Hydrostatic equilibrium... */
          ctl->hydz = scan_ctl(argc, argv, "HYDZ", -1, "-999", NULL);
04226
04227
04228
         /* Continua... */
         ctl->ctm_co2 = (int) scan_ctl(argc, argv, "CTM_CO2", -1, "1", NULL);
ctl->ctm_h2o = (int) scan_ctl(argc, argv, "CTM_H2O", -1, "1", NULL);
ctl->ctm_n2 = (int) scan_ctl(argc, argv, "CTM_N2", -1, "1", NULL);
ctl->ctm_o2 = (int) scan_ctl(argc, argv, "CTM_O2", -1, "1", NULL);
04229
04230
04231
04232
04233
04234
          /* Ray-tracing... */
         ctl->refrac = (int) scan_ctl(argc, argv, "REFRAC", -1, "1", NULL);
ctl->rayds = scan_ctl(argc, argv, "RAYDS", -1, "10", NULL);
04235
```

```
ctl->raydz = scan_ctl(argc, argv, "RAYDZ", -1, "0.5", NULL);
04238
           /* Field of view... */
04239
           scan_ctl(argc, argv, "FOV", -1, "-", ctl->fov);
04240
04241
04242
           /* Retrieval interface... */
           ctl->retp_zmin = scan_ctl(argc, argv, "RETP_ZMIN", -1, "-999", NULL);
ctl->retp_zmax = scan_ctl(argc, argv, "RETP_ZMAX", -1, "-999", NULL);
ctl->rett_zmin = scan_ctl(argc, argv, "RETT_ZMIN", -1, "-999", NULL);
04243
04244
04245
           ctl->rett_zmax = scan_ctl(argc, argv, "RETT_ZMAX", -1, "-999", NULL);
for (ig = 0; ig < ctl->ng; ig++) {
04246
04247
            ctl->retq_zmin[ig] = scan_ctl(argc, argv, "RETO_ZMIN", ig, "-999", NULL); ctl->retq_zmax[ig] = scan_ctl(argc, argv, "RETO_ZMAX", ig, "-999", NULL);
04248
04249
04250
04251
           for (iw = 0; iw < ctl->nw; iw++) {
           ctl->retk_zmin[iw] = scan_ctl(argc, argv, "RETK_ZMIN", iw, "-999", NULL);
ctl->retk_zmax[iw] = scan_ctl(argc, argv, "RETK_ZMAX", iw, "-999", NULL);
04252
04253
04254
04256
           /* Output flags... */
04257
           ctl->write_bbt = (int) scan_ctl(argc, argv, "WRITE_BBT", -1, "0", NULL);
04258
           ctl->write_matrix =
              (int) scan_ctl(argc, argv, "WRITE_MATRIX", -1, "0", NULL);
04259
04260 }
```

Here is the call graph for this function:



5.13.2.36 void read\_matrix ( const char \* dirname, const char \* filename, gsl\_matrix \* matrix )

Read matrix.

Definition at line 4264 of file jurassic.c.

```
04267
04268
04269
       FILE *in;
04270
       char dum[LEN], file[LEN], line[LEN];
04271
04272
04273
       double value;
04274
04275
       int i, j;
04276
04277
        /* Set filename... */
        if (dirname != NULL)
04278
04279
         sprintf(file, "%s/%s", dirname, filename);
04280
04281
         sprintf(file, "%s", filename);
04282
04283
       /* Write info... */
04284
       printf("Read matrix: %s\n", file);
04285
04286
04287
       if (!(in = fopen(file, "r")))
04288
         ERRMSG("Cannot open file!");
04289
04290
       /* Read data... */
04291
       gsl_matrix_set_zero(matrix);
04292
       while (fgets(line, LEN, in))
04293
         if (sscanf(line, "%d %s %s %s %s %s %d %s %s %s %s %s %lg",
04294
                    &i, dum, dum, dum, dum, dum,
                     &j, dum, dum, dum, dum, &value) == 13)
04295
04296
            gsl_matrix_set(matrix, (size_t) i, (size_t) j, value);
04297
04298
        /* Close file... */
04299
       fclose(in);
04300 }
```

5.13.2.37 void read\_obs ( const char \* dirname, const char \* filename, ctl\_t \* ctl, obs\_t \* obs )

Read observation data.

Definition at line 4304 of file jurassic.c.

```
04308
04309
04310
             FILE *in;
04312
             char file[LEN], line[LEN], *tok;
04313
04314
             int id;
04315
             /* Init... */
obs->nr = 0;
04316
04317
04318
04319
              /* Set filename... */
04320
              if (dirname != NULL)
                sprintf(file, "%s/%s", dirname, filename);
04321
04322
             else
04323
                 sprintf(file, "%s", filename);
04324
04325
              /\star Write info... \star/
04326
             printf("Read observation data: %s\n", file);
04327
04328
             /* Open file... */
if (!(in = fopen(file, "r")))
04329
04330
                 ERRMSG("Cannot open file!");
04331
04332
              /* Read line... */
             while (fgets(line, LEN, in)) {
04333
                /* Read data... */
TOK(line, tok, "%lg", obs->time[obs->nr]);
TOK(NULL, tok, "%lg", obs->obsz[obs->nr]);
TOK(NULL, tok, "%lg", obs->obslon[obs->nr]);
TOK(NULL, tok, "%lg", obs->obslat[obs->nr]);
TOK(NULL, tok, "%lg", obs->vpz[obs->nr]);
TOK(NULL, tok, "%lg", obs->vplon[obs->nr]);
TOK(NULL, tok, "%lg", obs->vplat[obs->nr]);
TOK(NULL, tok, "%lg", obs->tpz[obs->nr]);
TOK(NULL, tok, "%lg", obs->tplon[obs->nr]);
TOK(NULL, tok, "%lg", obs->tplon[obs->nr]);
TOK(NULL, tok, "%lg", obs->tplat[obs->nr]);
for (id = 0; id < ctl->nd; id++)
    TOK(NULL, tok, "%lg", obs->rad[id][obs->nr]);
for (id = 0; id < ctl->nd; id++)
    TOK(NULL, tok, "%lg", obs->tplat[id][obs->nr]);
04334
04335
04336
04337
04338
04339
04340
04341
04342
04343
04344
04345
04346
04347
04348
04349
04350
                 /* Increment counter... */
04351
04352
                 if ((++obs->nr) > NR)
                     ERRMSG("Too many rays!");
04353
04354
04355
04356
              /* Close file... */
04357
             fclose(in);
04358
04359
             /* Check number of points... */
if (obs->nr < 1)</pre>
04360
                 ERRMSG("Could not read any data!");
04361
04362 }
```

5.13.2.38 void read\_shape ( const char \* filename, double \* x, double \* y, int \* n )

Read shape function.

Definition at line 4366 of file jurassic.c.

```
04370 {
04371
04372 FILE *in;
04373
04374 char line[LEN];
04375
04376 /* Write info... */
04377 printf("Read shape function: %s\n", filename);
```

```
04378
04379
         /* Open file... */
         if (!(in = fopen(filename, "r")))
04380
          ERRMSG("Cannot open file!");
04381
04382
         /* Read data... */
04383
04384
        *n = 0;
04385
        while (fgets(line, LEN, in))
         if (sscanf(line, "%lg %lg", &x[*n], &y[*n]) == 2)
if ((++(*n)) > NSHAPE)
    ERRMSG("Too many data points!");
04386
04387
04388
04389
04390
        /* Check number of points... */
04391
04392
          ERRMSG("Could not read any data!");
04393
        /* Close file... */
04394
04395
        fclose(in);
04396 }
```

5.13.2.39 double refractivity ( double p, double t )

Compute refractivity (return value is n - 1).

Definition at line 4400 of file jurassic.c.

5.13.2.40 double scan\_ctl (int argc, char \* argv[], const char \* varname, int arridx, const char \* defvalue, char \* value)

Search control parameter file for variable entry.

Definition at line 4410 of file jurassic.c.

```
04416
                            {
04417
04418
          FILE *in = NULL;
04419
04420
          char dummy[LEN], fullname1[LEN], fullname2[LEN], line[LEN],
04421
            msg[LEN], rvarname[LEN], rval[LEN];
04422
04423
          int contain = 0, i;
04424
04425
          /* Open file... */
          if (argv[1][0] != '-')
04426
           if (!(in = fopen(argv[1], "r")))
    ERRMSG("Cannot open file!");
04427
04428
04429
04430
          /* Set full variable name... */
04431
          if (arridx >= 0) {
            sprintf(fullname1, "%s[%d]", varname, arridx);
sprintf(fullname2, "%s[*]", varname);
04432
04433
04434
          sprintf(fullname1, "%s", varname);
sprintf(fullname2, "%s", varname);
04435
04436
04437
04438
          /* Read data... */
04439
04440
          if (in != NULL)
            while (fgets(line, LEN, in))
  if (sscanf(line, "%s %s %s", rvarname, dummy, rval) == 3)
  if (strcasecmp(rvarname, fullname1) == 0 ||
    strcasecmp(rvarname, fullname2) == 0) {
04441
04442
04443
04444
04445
                     contain = 1;
04446
                     break;
04447
          for (i = 1; i < argc - 1; i++)
04448
           if (strcasecmp(argv[i], fullname1) == 0 ||
    strcasecmp(argv[i], fullname2) == 0) {
04449
04450
04451
                sprintf(rval, "%s", argv[i + 1]);
```

```
04452
           contain = 1;
04453
           break;
04454
04455
       /* Close file... */
04456
       if (in != NULL)
04457
         fclose(in);
04459
04460
       /* Check for missing variables... */
04461
        if (!contain) {
        if (strlen(defvalue) > 0)
04462
           sprintf(rval, "%s", defvalue);
04463
04464
04465
           sprintf(msg, "Missing variable %s!\n", fullname1);
04466
            ERRMSG(msg);
04467
04468
04469
04470
       /* Write info... */
04471
       printf("%s = %s\n", fullname1, rval);
04472
04473
        /* Return values... */
       if (value != NULL)
    sprintf(value, "%s", rval);
04474
04475
04476
       return atof(rval);
04477 }
```

5.13.2.41 void tangent\_point (  $los_t * los_t$ , double \*  $tpz_t$ , double \*  $tplon_t$ , double \*  $tplon_t$ )

Find tangent point of a given LOS.

Definition at line 4481 of file jurassic.c.

```
04485
04486
04487
          double a, b, c, dummy, v[3], v0[3], v2[3], x, x1, x2, yy0, yy1, yy2;
04488
04489
          size_t i, ip;
04490
04491
          /* Find minimum altitude... */
04492
          ip = gsl_stats_min_index(los->z, 1, (size_t) los->np);
04493
04494
          /* Nadir or zenith... */
         if (ip <= 0 || ip >= (size_t) los->np - 1) {
  *tpz = los->z[los->np - 1];
  *tplon = los->lon[los->np - 1];
04495
04496
04497
04498
            *tplat = los->lat[los->np - 1];
04499
04500
04501
          /* Limb... */
04502
          else {
04503
             /* Determine interpolating polynomial y=a*x^2+b*x+c... */
04505
            yy0 = los -> z[ip - 1];
             yy1 = los \rightarrow z[ip];
04506
             yy2 = los -> z[ip + 1];
04507
            yyz - 10S-2[1p + 1],

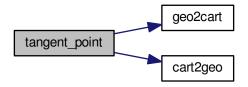
x1 = sqrt(gsl_pow_2(los->ds[ip]) - gsl_pow_2(yy1 - yy0));

x2 = x1 + sqrt(gsl_pow_2(los->ds[ip + 1]) - gsl_pow_2(yy2 - yy1));

a = 1 / (x1 - x2) * (-(yy0 - yy1) / x1 + (yy0 - yy2) / x2);

b = -(yy0 - yy1) / x1 - a * x1;
04508
04509
04510
04511
04512
04513
04514
             /* Get tangent point location... */
            x = -b / (2 * a);
*tpz = a * x * x + b * x + c;
04515
04517
             geo2cart(los->z[ip - 1], los->lon[ip - 1], los->lat[ip - 1], v0);
04518
             geo2cart(los->z[ip + 1], los->lon[ip + 1], los->lat[ip + 1], v2);
             for (i = 0; i < 3; i++)
v[i] = LIN(0.0, v0[i], x2, v2[i], x);
04519
04520
04521
            cart2geo(v, &dummy, tplon, tplat);
04522
04523 }
```

Here is the call graph for this function:



5.13.2.42 void time2jsec (int year, int mon, int day, int hour, int min, int sec, double remain, double \* jsec )

Convert date to seconds.

Definition at line 4527 of file jurassic.c.

```
04535
04537
        struct tm t0, t1;
04538
04539
        t0.tm_year = 100;
        t0.tm_mon = 0;
04540
        t0.tm_mday = 1;
04541
04542
        t0.tm\_hour = 0;
        t0.tm_min = 0;
t0.tm_sec = 0;
04543
04544
04545
        t1.tm_year = year - 1900;
t1.tm_mon = mon - 1;
04546
04547
04548
        t1.tm_mday = day;
        t1.tm_hour = hour;
04549
04550
        t1.tm_min = min;
04551
        t1.tm_sec = sec;
04552
        *jsec = (double) timegm(&t1) - (double) timegm(&t0) + remain;
04553
04554 }
```

5.13.2.43 void timer ( const char \* name, const char \* file, const char \* func, int line, int mode )

Measure wall-clock time.

Definition at line 4558 of file jurassic.c.

```
04563
04564
        static double dt_w, w0[10];
04565
04566
04567
        static int 10[10], nt;
04568
04569
        struct timeval tim;
04570
04571
         /* Start new timer... */
04572
        if (mode == 1) {
          gettimeofday(&tim, NULL);
          w0[nt] = (double) tim.tv_sec + (double) tim.tv_usec / 1e6;
10[nt] = line;
04574
04575
           if ((++nt) >= 10)
    ERRMSG("Too many timers!");
04576
04577
04578
04579
04580
        /* Write elapsed time... */
```

```
04581
       else {
04582
04583
          /\star Check timer index... \star/
          if (nt - 1 < 0)
04584
           ERRMSG("Coding error!");
04585
04586
04587
          /* Get time differences... */
04588
          gettimeofday(&tim, NULL);
04589
          dt_w = (double) tim.tv_sec + (double) tim.tv_usec / 1e6 - w0[nt - 1];
04590
04591
          /* Write elapsed time... */
         printf("Timer '%s' (%s, %s, 1%d-%d): %.3f sec\n",
04592
                 name, file, func, 10[nt - 1], line, dt_w);
04593
04594
04595
04596
        /* Stop timer... */
        if (mode == 3)
04597
04598
          nt--;
04599 }
```

5.13.2.44 void write\_atm ( const char \* dirname, const char \* filename, ctl\_t \* ctl, atm\_t \* atm )

Write atmospheric data.

Definition at line 4603 of file jurassic.c.

```
04607
                             {
04608
          FILE *out;
04609
04610
04611
          char file[LEN];
04612
04613
          int iq, ip, iw, n = 6;
04614
04615
           /* Set filename... */
04616
          if (dirname != NULL)
04617
             sprintf(file, "%s/%s", dirname, filename);
04618
          else
04619
             sprintf(file, "%s", filename);
04620
04621
          /* Write info... */
04622
          printf("Write atmospheric data: %s\n", file);
04624
           /* Create file... */
          if (!(out = fopen(file, "w")))
04625
             ERRMSG("Cannot create file!");
04626
04627
04628
           /* Write header... */
04629
04630
                      "# $1 = time (seconds since 2000-01-01T00:00Z) \n"
                      "# $2 = altitude [km] \n"
04631
                      "# $3 = longitude [deg]\n"
04632
                      "# $4 = latitude [deg]\n"
04633
                      "# $5 = pressure [hPa]\n" "# $6 = temperature [K]\n");
04634
          for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, "# $%d = %s volume mixing ratio\n", ++n, ctl->emitter[ig]);
04636
          for (iw = 0; iw < ctl->nw; iw++)
  fprintf(out, "# $%d = window %d: extinction [1/km]\n", ++n, iw);
04637
04638
04639
04640
           /* Write data... */
          for (ip = 0; ip < atm->np; ip++) {
04641
            for (ip = 0; ip < atm->np; ip++) {
    if (ip == 0 || atm->lat[ip] != atm->lat[ip - 1]
        || atm->lon[ip] != atm->lon[ip - 1])
    fprintf(out, "\n");
    fprintf(out, "%.2f %g %g %g %g", atm->time[ip], atm->z[ip],
        atm->lon[ip], atm->lat[ip], atm->p[ip], atm->t[ip]);
    for (iz = 0. iz < ctl->ng. ig++)
04643
04644
04645
04646
             for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, " %g", atm->q[ig][ip]);
04647
04648
             for (iw = 0; iw < ctl->nw; iw++)
  fprintf(out, " %g", atm->k[iw][ip]);
fprintf(out, "\n");
04649
04650
04651
04652
04653
           /* Close file... */
04654
04655
          fclose(out);
04656 }
```

5.13.2.45 void write\_matrix ( const char \* dirname, const char \* filename, ctl\_t \* ctl, gsl\_matrix \* matrix, atm\_t \* atm, obs\_t \* obs, const char \* rowspace, const char \* colspace, const char \* sort )

Write matrix.

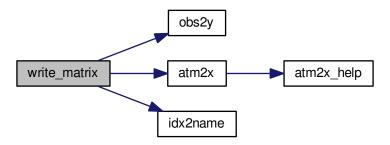
Definition at line 4660 of file jurassic.c.

```
04669
04670
04671
        FILE *out;
04672
04673
       char file[LEN], quantity[LEN];
04674
        int *cida, *ciqa, *cipa, *cira, *rida, *riqa, *ripa, *rira;
04676
04677
       size_t i, j, nc, nr;
04678
04679
        /* Check output flag... */
04680
        if (!ctl->write_matrix)
04681
         return;
04682
04683
        /* Allocate... */
04684
       ALLOC(cida, int, M);
04685
        ALLOC(ciqa, int,
04686
             N);
04687
        ALLOC(cipa, int,
04688
             N);
04689
        ALLOC(cira, int,
04690
             M);
        ALLOC(rida, int,
04691
04692
             M);
04693
        ALLOC(riqa, int,
04694
              N);
04695
        ALLOC(ripa, int,
04696
             N);
        ALLOC(rira, int,
04697
04698
             M);
04699
04700
        /* Set filename... */
04701
        if (dirname != NULL)
04702
         sprintf(file, "%s/%s", dirname, filename);
04703
        else
          sprintf(file, "%s", filename);
04704
04705
04706
        /* Write info... */
04707
       printf("Write matrix: %s\n", file);
04708
04709
        /* Create file... */
       if (!(out = fopen(file, "w")))
04710
04711
         ERRMSG("Cannot create file!");
04712
04713
        /* Write header (row space)... */
04714
       if (rowspace[0] == 'y') {
04715
04716
          fprintf(out.
04717
                   "# $1 = Row: index (measurement space) \n"
04718
                  "# $2 = Row: channel wavenumber [cm^-1]\n"
04719
                  "# $3 = \text{Row: time (seconds since } 2000-01-01T00:00Z) \n"
04720
                  "# $4 = Row: view point altitude [km]\n"
04721
                  "# $5 = Row: view point longitude [deg] \n"
                  "# $6 = Row: view point latitude [deg]\n");
04722
04723
04724
          /* Get number of rows... */
04725
          nr = obs2y(ctl, obs, NULL, rida, rira);
04726
04727
       } else {
04728
04729
          fprintf(out,
04730
                   "# $1 = Row: index (state space) \n"
04731
                  "# $2 = Row: name of quantity\n"
04732
                  "# $3 = Row: time (seconds since 2000-01-01T00:00Z)\n"
04733
                  "# $4 = Row: altitude [km]\n"
                  "# $5 = Row: longitude [deg]\n" "# $6 = Row: latitude [deg]\n");
04734
04735
04736
          /* Get number of rows... */
         nr = atm2x(ctl, atm, NULL, riqa, ripa);
04737
04738
04739
04740
        /\star Write header (column space)... \star/
04741
        if (colspace[0] == 'y') {
04742
04743
          fprintf(out,
04744
                  "# $7 = Col: index (measurement space) \n"
```

```
04745
                   "# $8 = Col: channel wavenumber [cm^-1]\n"
04746
                   "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04747
                    "# $10 = Col: view point altitude [km] \n"
04748
                   "# $11 = Col: view point longitude [deg]\n"
                   "# $12 = Col: view point latitude [deg]\n");
04749
04750
04751
           /* Get number of columns... */
04752
          nc = obs2y(ctl, obs, NULL, cida, cira);
04753
04754
        } else {
04755
04756
          fprintf(out,
                    "# $7 = Col: index (state space)\n"
04757
04758
                   "# $8 = Col: name of quantity\n"
04759
                   "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04760
                   "# $10 = Col: altitude [km] \n"
                   "# \$11 = Col: longitude [deg]\n" "# \$12 = Col: latitude [deg]\n");
04761
04762
04763
           /\star Get number of columns... \star/
04764
          nc = atm2x(ctl, atm, NULL, ciqa, cipa);
04765
04766
        /* Write header entry... */ fprintf(out, "# $13 = Matrix element\n\n");
04767
04768
04769
04770
        /* Write matrix data... */
04771
        i = j = 0;
04772
        while (i < nr && j < nc) {</pre>
04773
04774
           /* Write info about the row... */
          04775
04777
04778
                     obs->time[rira[i]], obs->vpz[rira[i]],
04779
                     obs->vplon[rira[i]], obs->vplat[rira[i]]);
04780
           else {
04781
             idx2name(ctl, riqa[i], quantity);
fprintf(out, "%d %s %.2f %g %g %g", (int) i, quantity,
04782
04783
                     atm->time[ripa[i]], atm->z[ripa[i]],
04784
                     atm->lon[ripa[i]], atm->lat[ripa[i]]);
04785
04786
          /* Write info about the column... */
if (colspace[0] == 'y')
  fprintf(out, " %d %g %.2f %g %g %g",
04787
04788
04789
04790
                     (int) j, ctl->nu[cida[j]],
04791
                     obs->time[cira[j]], obs->vpz[cira[j]],
04792
                     obs->vplon[cira[j]], obs->vplat[cira[j]]);
04793
          else {
            idx2name(ctl, ciqa[j], quantity);
fprintf(out, " %d %s %.2f %g %g %g", (int) j, quantity,
04794
04795
04796
                     atm->time[cipa[j]], atm->z[cipa[j]]
04797
                     atm->lon[cipa[j]], atm->lat[cipa[j]]);
04798
          }
04799
          /* Write matrix entry... */ fprintf(out, " g^n, gsl_matrix_get(matrix, i, j));
04800
04801
04802
04803
           /* Set matrix indices... */
04804
           if (sort[0] == 'r') {
04805
             j++;
             if (j >= nc) {
04806
04807
              j = 0;
04808
04809
               fprintf(out, "\n");
04810
04811
           } else {
04812
             i++;
04813
             if (i >= nr) {
              i = 0;
04814
04815
               j++;
04816
               fprintf(out, "\n");
04817
            }
04818
          }
04819
        }
04820
04821
         /* Close file... */
04822
        fclose(out);
04823
04824
        /* Free... */
04825
        free(cida);
04826
        free (ciqa);
04827
        free(cipa);
04828
        free (cira);
04829
        free(rida);
04830
        free (riga);
04831
        free (ripa):
```

```
04832 free(rira);
04833 }
```

Here is the call graph for this function:



5.13.2.46 void write\_obs ( const char \* dirname, const char \* filename, ctl t \* ctl, obs t \* obs)

Write observation data.

Definition at line 4837 of file jurassic.c.

```
04841
04842
04843
        FILE *out;
04844
04845
        char file[LEN];
04846
04847
        int id, ir, n = 10;
04848
04849
        /* Set filename... */
04850
        if (dirname != NULL)
04851
          sprintf(file, "%s/%s", dirname, filename);
04852
          sprintf(file, "%s", filename);
04853
04854
04855
        /* Write info... */
04856
        printf("Write observation data: %s\n", file);
04857
04858
        /* Create file... */
        if (!(out = fopen(file, "w")))
04859
04860
          ERRMSG("Cannot create file!");
04861
04862
        /* Write header... */
04863
        fprintf(out,
04864
                 "# $1 = time (seconds since 2000-01-01T00:00Z) \n"
                 "# $2 = observer altitude [km] \n"
04865
                 "# $3 = observer longitude [deg]\n"
04866
                 "# $4 = observer latitude [deg]\n'
04867
                 "# $5 = view point altitude [km]\n"
04868
                 "# $6 = view point longitude [deg]\n"
04869
04870
                 "# $7 = view point latitude [deg]\n"
                 "# $8 = tangent point altitude [km]\n" "# $9 = tangent point longitude [deg]\n"
04871
04872
                 "# $10 = tangent point latitude [deg]\n");
04873
04874
        for (id = 0; id < ctl->nd; id++)
04875
         fprintf(out, "# $%d = channel %g: radiance [W/(m^2 sr cm^-1)]\n", ++n, ctl->nu[id]);
04876
        for (id = 0; id < ctl->nd; id+)
  fprintf(out, "# $%d = channel %g: transmittance\n", ++n, ctl->nu[id]);
04877
04878
04879
04880
        /* Write data... */
04881
        for (ir = 0; ir < obs->nr; ir++) {
04882
          if (ir == 0 || obs->time[ir] != obs->time[ir - 1])
```

```
fprintf(out, "\n");
fprintf(out, "%.2f %g %g %g %g %g %g %g %g %g %g", obs->time[ir],
04883
04884
04885
                         obs->obsz[ir], obs->obslon[ir], obs->obslat[ir],
                        obs->vpz[ir], obs->vplon[ir], obs->vplat[ir],
04886
             obs->tpz[ir], obs->tplon[ir], obs->tplat[ir]);
for (id = 0; id < ctl->nd; id++)
  fprintf(out, " %g", obs->rad[id][ir]);
04887
04888
04889
             for (id = 0; id < ctl->nd; id++)
  fprintf(out, " %g", obs->tau[id][ir]);
fprintf(out, "\n");
04890
04891
04892
04893
04894
04895
           /* Close file... */
04896
          fclose(out);
04897 }
```

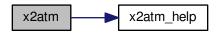
5.13.2.47 void x2atm ( ctl\_t \* ctl, gsl\_vector \* x, atm\_t \* atm )

Decompose parameter vector or state vector.

Definition at line 4901 of file jurassic.c.

```
04904
04905
04906
       int iq, iw;
04907
       size_t n = 0;
04908
04909
04910
       /* Set pressure... */
       x2atm_help(atm, ctl->retp_zmin, ctl->retp_zmax, atm->
04911
     p, x, &n);
04912
04913
        /* Set temperature... */
04914
       x2atm_help(atm, ctl->rett_zmin, ctl->rett_zmax, atm->
     t, x, &n);
04915
04916
       /* Set volume mixing ratio... */
       for (ig = 0; ig < ctl->ng; ig++)
04917
04918
         x2atm_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
04919
                     atm->q[ig], x, &n);
04920
04921
       /* Set extinction... */
04922
       for (iw = 0; iw < ctl->nw; iw++)
         x2atm_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
04923
04924
                     atm->k[iw], x, &n);
04925 }
```

Here is the call graph for this function:



5.13.2.48 void x2atm\_help ( atm\_t \* atm, double zmin, double zmax, double \* value, gsl\_vector \* x, size\_t \* n )

Extract elements from state vector.

Definition at line 4929 of file jurassic.c.

```
04935
04936
04937
        int ip;
04938
04939
        /* Extract state vector elements... */
04940
        for (ip = 0; ip < atm->np; ip++)
         if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {</pre>
04941
04942
            value[ip] = gsl_vector_get(x, *n);
04943
            (*n)++;
04944
04945 }
```

```
5.13.2.49 void y2obs ( ctl_t * ctl, gsl_vector * y, obs_t * obs )
```

Decompose measurement vector.

Definition at line 4949 of file jurassic.c.

```
04952
04953
04954
       int id, ir;
04955
04956
       size_t m = 0;
04958
        /* Decompose measurement vector... */
04959
        for (ir = 0; ir < obs->nr; ir++)
         for (id = 0; id < ctl->nd; id++)
04960
           if (gsl_finite(obs->rad[id][ir])) {
04961
04962
             obs->rad[id][ir] = gsl_vector_get(y, m);
04963
             m++;
04964
04965 }
```

```
00001 /*
00002
        This file is part of JURASSIC.
00003
00004
        JURASSIC is free software: you can redistribute it and/or modify
00005
        it under the terms of the GNU General Public License as published by
00006
        the Free Software Foundation, either version 3 of the License, or
00007
        (at your option) any later version.
80000
        JURASSIC is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00009
00010
00011
00012
        GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
       along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00028
00029 size_t atm2x(
        ctl_t * ctl,
atm_t * atm,
00030
00031
00032
        gsl\_vector * x,
00033
       int *iqa,
int *ipa) {
00034
00035
00036
       int ig, iw;
00037
00038
       size_t n = 0;
00039
00040
        /* Add pressure... */
00041
       atm2x_help(atm, ctl->retp_zmin, ctl->retp_zmax,
00042
                   atm->p, IDXP, x, iqa, ipa, &n);
00043
00044
        /* Add temperature... */
        atm2x_help(atm, ctl->rett_zmin, ctl->rett_zmax,
00045
00046
                   atm->t, IDXT, x, iqa, ipa, &n);
00047
00048
        /* Add volume mixing ratios...
00049
        for (ig = 0; ig < ctl->ng; ig++)
00050
        atm2x_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
00051
                     atm->q[ig], IDXQ(ig), x, iqa, ipa, &n);
00052
00053
        /* Add extinction... */
        for (iw = 0; iw < ctl->nw; iw++)
  atm2x_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
00054
00055
00056
                     atm->k[iw], IDXK(iw), x, iqa, ipa, &n);
00057
00058
        return n;
00059 }
```

```
00062
00063 void atm2x_help(
00064
         atm_t * atm,
00065
         double zmin,
00066
         double zmax,
00067
         double *value.
         int val_iqa,
00069
         gsl_vector
         int *iqa,
00070
00071
         int *ipa,
00072
         size_t * n) {
00073
00074
         int ip;
00075
00076
         /* Add elements to state vector... */
         for (ip = 0; ip < atm->np; ip++)
  if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {</pre>
00077
00078
00079
             if (x != NULL)
                gsl_vector_set(x, *n, value[ip]);
              if (iqa != NULL)
00081
00082
                iqa[*n] = val_iqa;
00083
              if (ipa != NULL)
                ipa[*n] = ip;
00084
00085
              (*n)++;
00086
00087 }
00088
00090
00091 double brightness (
00092
        double rad,
00093
         double nu) {
00094
00095
         return C2 * nu / gsl_log1p(C1 * gsl_pow_3(nu) / rad);
00096 }
00097
00098
00100
00101 void cart2geo(
00102
         double *x,
00103
         double *z,
         double *lon.
00104
00105
         double *lat) {
00106
00107
         double radius;
00108
         radius = NORM(x);
00109
         *lat = asin(x[2] / radius) * 180 / M_PI;
*lon = atan2(x[1], x[0]) * 180 / M_PI;
00110
00111
00112
         *z = radius - RE;
00113 }
00114
00116
00117 void climatology (
00118
        ctl_t * ctl,
         atm_t * atm) {
00119
00120
         static double z[121] = {
00121
           0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,
00122
00123
00124
            56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73,
00125
00126
            74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91,
00127
           92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107,
00128
           108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120
00129
00130
         static double pre[121] = {
00132
           1017, 901.083, 796.45, 702.227, 617.614, 541.644, 473.437, 412.288,
00133
            357.603, 308.96, 265.994, 228.348, 195.619, 167.351, 143.039, 122.198,
           104.369, 89.141, 76.1528, 65.0804, 55.641, 47.591, 40.7233, 34.8637, 29.8633, 25.5956, 21.9534, 18.8445, 16.1909, 13.9258, 11.9913, 10.34, 8.92988, 7.72454, 6.6924, 5.80701, 5.04654, 4.39238, 3.82902, 3.34337, 2.92413, 2.56128, 2.2464, 1.97258, 1.73384, 1.52519, 1.34242,
00134
00135
00136
00138
            1.18197, 1.04086, 0.916546, 0.806832, 0.709875, 0.624101, 0.548176,
           0.480974, 0.421507, 0.368904, 0.322408, 0.281386, 0.245249, 0.213465, 0.185549, 0.161072, 0.139644, 0.120913, 0.104568, 0.0903249, 0.0779269,
00139
00140
           0.0671493, 0.0577962, 0.0496902, 0.0426736, 0.0366093, 0.0313743, 0.0268598, 0.0229699, 0.0196206, 0.0167399, 0.0142646, 0.0121397,
00141
00142
            0.0103181, 0.00875775, 0.00742226, 0.00628076, 0.00530519, 0.00447183,
            0.00376124,\ 0.00315632,\ 0.00264248,\ 0.00220738,\ 0.00184003,\ 0.00153095,
00144
00145
            0.00127204, 0.00105608, 0.000876652, 0.00072798, 0.00060492,
           0.000503201, 0.000419226, 0.000349896, 0.000292659, 0.000245421, 0.000206394, 0.000174125, 0.000147441, 0.000125333, 0.000106985, 9.173e-05, 7.90172e-05, 6.84172e-05, 5.95574e-05, 5.21183e-05,
00146
00147
00148
```

```
4.58348e-05, 4.05127e-05, 3.59987e-05, 3.21583e-05, 2.88718e-05,
            2.60322e-05, 2.35687e-05, 2.14263e-05, 1.95489e-05
00150
00151
00152
00153
          static double tem[121] = { 285.14, 279.34, 273.91, 268.3, 263.24, 256.55, 250.2, 242.82, 236.17,
00154
             229.87, 225.04, 221.19, 218.85, 217.19, 216.26, 215.68, 215.42, 215.55, 215.92, 216.4, 216.93, 217.45, 218, 218.68, 219.39, 220.25, 221.3,
00156
00157
             222.41, 223.88, 225.42, 227.2, 229.52, 231.89, 234.51, 236.85, 239.42,
             241.94, 244.57, 247.36, 250.32, 253.34, 255.82, 258.27, 260.39, 262.03, 263.45, 264.2, 264.78, 264.67, 264.38, 263.24, 262.03, 260.02, 258.09, 255.63, 253.28, 250.43, 247.81, 245.26, 242.77, 240.38, 237.94, 235.79, 233.53, 231.5, 229.53, 227.6, 225.62, 223.77, 222.06, 220.33, 218.69, 217.18, 215.64, 214.13, 212.52, 210.86, 209.25,
00158
00159
00160
00161
00162
00163
             207.49, 205.81, 204.11, 202.22, 200.32, 198.39, 195.92, 193.46,
            190.94, 188.31, 185.82, 183.57, 181.43, 179.74, 178.64, 178.1, 178.25, 178.7, 179.41, 180.67, 182.31, 184.18, 186.6, 189.53, 192.66, 196.54, 201.13, 205.93, 211.73, 217.86, 225, 233.53, 242.57, 252.14, 261.48, 272.97, 285.26, 299.12, 312.2, 324.17, 338.34, 352.56, 365.28
00164
00165
00166
00168
00169
00170
          static double c2h2[121] = {
            1.352e-09, 2.83e-10, 1.269e-10, 6.926e-11, 4.346e-11, 2.909e-11,
00171
             2.014e-11, 1.363e-11, 8.71e-12, 5.237e-12, 2.718e-12, 1.375e-12, 5.786e-13, 2.16e-13, 7.317e-14, 2.551e-14, 1.055e-14, 4.758e-15,
00172
00173
             2.056e-15, 7.703e-16, 2.82e-16, 1.035e-16, 4.382e-17, 1.946e-17, 9.638e-18, 5.2e-18, 2.811e-18, 1.494e-18, 7.925e-19, 4.213e-19,
00174
00175
00176
             1.998e-19, 8.78e-20, 3.877e-20, 1.728e-20, 7.743e-21, 3.536e-21,
             1.623e-21, 7.508e-22, 3.508e-22, 1.65e-22, 7.837e-23, 3.733e-23, 1.808e-23, 8.77e-24, 4.285e-24, 2.095e-24, 1.032e-24, 5.082e-25,
00177
00178
00179
             2.506e-25, 1.236e-25, 6.088e-26, 2.996e-26, 1.465e-26, 0, 0, 0,
00180
             00181
00182
            00183
00184
          static double c2h6[121] = { 2.667e-09, 2.02e-09, 1.658e-09, 1.404e-09, 1.234e-09, 1.109e-09,
00185
00187
             1.012e-09, 9.262e-10, 8.472e-10, 7.71e-10, 6.932e-10, 6.216e-10,
             5.503e-10, 4.87e-10, 4.342e-10, 3.861e-10, 3.347e-10, 2.772e-10,
00188
00189
             2.209e-10, 1.672e-10, 1.197e-10, 8.536e-11, 5.783e-11, 3.846e-11,
             2.495e-11, 1.592e-11, 1.017e-11, 6.327e-12, 3.895e-12, 2.403e-12, 1.416e-12, 8.101e-13, 4.649e-13, 2.686e-13, 1.557e-13, 9.14e-14, 5.386e-14, 3.19e-14, 1.903e-14, 1.14e-14, 6.875e-15, 4.154e-15,
00190
00191
00192
             2.538e-15, 1.553e-15, 9.548e-16, 5.872e-16, 3.63e-16, 2.244e-16,
00193
00194
             1.388e-16, 8.587e-17, 5.308e-17, 3.279e-17, 2.017e-17, 1.238e-17,
             7.542e-18, 4.585e-18, 2.776e-18, 1.671e-18, 9.985e-19, 5.937e-19, 3.518e-19, 2.07e-19, 1.215e-19, 7.06e-20, 4.097e-20, 2.37e-20, 1.363e-20, 7.802e-21, 4.441e-21, 2.523e-21, 1.424e-21, 8.015e-22,
00195
00196
00197
00198
             4.497e-22, 2.505e-22, 1.391e-22, 7.691e-23, 4.238e-23, 2.331e-23,
             1.274e-23, 6.929e-24, 3.752e-24, 2.02e-24, 1.083e-24, 5.774e-25,
             3.041e-25, 1.593e-25, 8.308e-26, 4.299e-26, 2.195e-26, 1.112e-26,
00200
00201
             00202
            0, 0, 0, 0, 0, 0, 0, 0
00203
00204
          static double ccl4[121] = {
            1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10,
00206
00207
             1.075e-10, 1.075e-10, 1.075e-10, 1.06e-10, 1.024e-10, 9.69e-11,
             8.93e-11, 8.078e-11, 7.213e-11, 6.307e-11, 5.383e-11, 4.49e-11,
00208
             3.609e-11, 2.705e-11, 1.935e-11, 1.385e-11, 8.35e-12, 5.485e-12, 3.853e-12, 2.22e-12, 5.875e-13, 3.445e-13, 1.015e-13, 6.075e-14, 4.383e-14, 2.692e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00209
00210
             le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14,
00212
00213
             1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00214
             1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00215
             1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
             1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00216
             1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00218
             le-14, le-14,
             le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14, le-14,
00219
00220
             1e-14, 1e-14, 1e-14
00221
          };
00222
00223
          static double ch4[121] = {
            1.864e-06, 1.835e-06, 1.819e-06, 1.805e-06, 1.796e-06, 1.788e-06,
             1.782e-06, 1.776e-06, 1.769e-06, 1.761e-06, 1.749e-06, 1.734e-06,
00225
00226
             1.716e-06, 1.692e-06, 1.654e-06, 1.61e-06, 1.567e-06, 1.502e-06,
            1.438-06, 1.371e-06, 1.323e-06, 1.277e-06, 1.232e-06, 1.188e-06, 1.147e-06, 1.108e-06, 1.07e-06, 1.027e-06, 9.854e-07, 9.416e-07, 8.933e-07, 8.478e-07, 7.988e-07, 7.515e-07, 7.07e-07, 6.64e-07, 6.239e-07, 5.864e-07, 5.512e-07, 5.184e-07, 4.87e-07, 4.571e-07,
00227
00228
00229
             4.296e-07, 4.04e-07, 3.802e-07, 3.578e-07, 3.383e-07, 3.203e-07, 3.032e-07, 2.889e-07, 2.76e-07, 2.635e-07, 2.519e-07, 2.409e-07,
00231
00232
00233
             2.302e-07, 2.219e-07, 2.144e-07, 2.071e-07, 1.999e-07, 1.93e-07,
             1.862e-07, 1.795e-07, 1.731e-07, 1.668e-07, 1.607e-07, 1.548e-07, 1.49e-07, 1.434e-07, 1.38e-07, 1.328e-07, 1.277e-07, 1.227e-07,
00234
00235
```

```
1.18e-07, 1.134e-07, 1.089e-07, 1.046e-07, 1.004e-07, 9.635e-08,
             9.245e-08, 8.867e-08, 8.502e-08, 8.15e-08, 7.809e-08, 7.48e-08, 7.159e-08, 6.849e-08, 6.55e-08, 6.262e-08, 5.98e-08, 5.708e-08,
00237
00238
00239
             5.448e-08, 5.194e-08, 4.951e-08, 4.72e-08, 4.5e-08, 4.291e-08,
             4.093e-08, 3.905e-08, 3.729e-08, 3.563e-08, 3.408e-08, 3.265e-08, 3.128e-08, 2.996e-08, 2.87e-08, 2.76e-08, 2.657e-08, 2.558e-08,
00240
00241
              2.467e-08, 2.385e-08, 2.307e-08, 2.234e-08, 2.168e-08, 2.108e-08,
00243
             2.05e-08, 1.998e-08, 1.947e-08, 1.902e-08, 1.86e-08, 1.819e-08,
00244
             1.782e-08
00245
00246
00247
          static double clo[121] = {
00248
             7.419e-15, 1.061e-14, 1.518e-14, 2.195e-14, 3.175e-14, 4.666e-14,
             6.872e-14, 1.03e-13, 1.553e-13, 2.375e-13, 3.664e-13, 5.684e-13,
00249
00250
             8.915e-13, 1.402e-12, 2.269e-12, 4.125e-12, 7.501e-12, 1.257e-11,
             2.048e-11, 3.338e-11, 5.44e-11, 8.846e-11, 1.008e-10, 1.082e-10, 1.157e-10, 1.232e-10, 1.312e-10, 1.539e-10, 1.822e-10, 2.118e-10,
00251
00252
             2.387e-10, 2.687e-10, 2.875e-10, 3.031e-10, 3.23e-10, 3.648e-10, 4.117e-10, 4.477e-10, 4.633e-10, 4.794e-10, 4.95e-10, 5.104e-10,
00253
             5.259e-10, 5.062e-10, 4.742e-10, 4.443e-10, 4.051e-10, 3.659e-10,
00255
00256
             3.305e-10, 2.911e-10, 2.54e-10, 2.215e-10, 1.927e-10, 1.675e-10,
00257
             1.452e-10, 1.259e-10, 1.09e-10, 9.416e-11, 8.119e-11, 6.991e-11,
             6.015e-11, 5.163e-11, 4.43e-11, 3.789e-11, 3.24e-11, 2.769e-11, 2.361e-11, 2.011e-11, 1.71e-11, 1.453e-11, 1.233e-11, 1.045e-11, 8.851e-12, 7.48e-12, 6.316e-12, 5.326e-12, 4.487e-12, 3.778e-12,
00258
00259
00260
             3.176e-12, 2.665e-12, 2.234e-12, 1.87e-12, 1.563e-12, 1.304e-12,
             1.085e-12, 9.007e-13, 7.468e-13, 6.179e-13, 5.092e-13, 4.188e-13,
00262
00263
             3.442e-13, 2.816e-13, 2.304e-13, 1.885e-13, 1.542e-13, 1.263e-13,
             1.035e-13, 8.5e-14, 7.004e-14, 5.783e-14, 4.795e-14, 4.007e-14,
00264
             3.345e-14, 2.792e-14, 2.33e-14, 1.978e-14, 1.686e-14, 1.438e-14, 1.234e-14, 1.07e-14, 9.312e-15, 8.131e-15, 7.164e-15, 6.367e-15, 5.67e-15, 5.088e-15, 4.565e-15, 4.138e-15, 3.769e-15, 3.432e-15,
00265
00266
00267
             3.148e-15
00268
00269
00270
          static double clono2[121] = {
00271
            1.011e-13, 1.515e-13, 2.272e-13, 3.446e-13, 5.231e-13, 8.085e-13, 1.253e-12, 1.979e-12, 3.149e-12, 5.092e-12, 8.312e-12, 1.366e-11,
00272
00274
             2.272e-11, 3.791e-11, 6.209e-11, 9.101e-11, 1.334e-10,
00275
             2.853e-10, 3.94e-10, 4.771e-10, 5.771e-10, 6.675e-10, 7.665e-10,
00276
             8.504e-10, 8.924e-10, 9.363e-10, 8.923e-10, 8.411e-10, 7.646e-10,
             6.525e-10, 5.576e-10, 4.398e-10, 3.403e-10, 2.612e-10, 1.915e-10, 1.407e-10, 1.028e-10, 7.455e-11, 5.42e-11, 3.708e-11, 2.438e-11, 1.618e-11, 1.075e-11, 7.17e-12, 4.784e-12, 3.205e-12, 2.147e-12,
00277
00278
00279
             1.44e-12, 9.654e-13, 6.469e-13, 4.332e-13, 2.891e-13, 1.926e-13,
00280
00281
             1.274e-13, 8.422e-14, 5.547e-14, 3.636e-14, 2.368e-14, 1.536e-14,
00282
             9.937e-15, 6.39e-15, 4.101e-15, 2.61e-15, 1.659e-15, 1.052e-15,
             6.638e-16, 4.172e-16, 2.61e-16, 1.63e-16, 1.013e-16, 6.275e-17, 3.879e-17, 2.383e-17, 1.461e-17, 8.918e-18, 5.43e-18, 3.301e-18,
00283
00284
             1.997e-18, 1.203e-18, 7.216e-19, 4.311e-19, 2.564e-19, 1.519e-19,
00285
             8.911e-20, 5.203e-20, 3.026e-20, 1.748e-20, 9.99e-21, 5.673e-21,
             3.215e-21, 1.799e-21, 1.006e-21, 5.628e-22, 3.146e-22, 1.766e-22,
00287
00288
             9.94e-23, 5.614e-23, 3.206e-23, 1.841e-23, 1.071e-23, 6.366e-24,
             3.776e-24, 2.238e-24, 1.326e-24, 8.253e-25, 5.201e-25, 3.279e-25, 2.108e-25, 1.395e-25, 9.326e-26, 6.299e-26, 4.365e-26, 3.104e-26, 2.219e-26, 1.621e-26, 1.185e-26, 8.92e-27, 6.804e-27, 5.191e-27,
00289
00290
00291
00292
00293
00294
00295
           static double co[121] = {
             1.907e-07, 1.553e-07, 1.362e-07, 1.216e-07, 1.114e-07, 1.036e-07, 9.737e-08, 9.152e-08, 8.559e-08, 7.966e-08, 7.277e-08, 6.615e-08, 5.884e-08, 5.22e-08, 4.699e-08, 4.284e-08, 3.776e-08, 3.274e-08,
00296
00297
             2.845e-08, 2.479e-08, 2.246e-08, 2.054e-08, 1.991e-08, 1.951e-08,
00299
00300
             1.94e-08, 2.009e-08, 2.1e-08, 2.201e-08, 2.322e-08, 2.45e-08,
00301
             2.602e-08, 2.73e-08, 2.867e-08, 2.998e-08, 3.135e-08, 3.255e-08,
00302
             3.352e-08, 3.426e-08, 3.484e-08, 3.53e-08, 3.593e-08, 3.671e-08, 3.759e-08, 3.945e-08, 4.192e-08, 4.49e-08, 5.03e-08, 5.703e-08,
00303
             6.538e-08, 7.878e-08, 9.644e-08, 1.196e-07, 1.498e-07, 1.904e-07,
00304
00305
              2.422e-07, 3.055e-07, 3.804e-07, 4.747e-07, 5.899e-07, 7.272e-07,
             8.91e-07, 1.071e-06, 1.296e-06, 1.546e-06, 1.823e-06, 2.135e-06, 2.44e-06, 2.714e-06, 2.967e-06, 3.189e-06, 3.391e-06, 3.58e-06,
00306
00307
             3.773e-06, 4.022e-06, 4.346e-06, 4.749e-06, 5.199e-06, 5.668e-06, 6.157e-06, 6.688e-06, 7.254e-06, 7.867e-06, 8.539e-06, 9.26e-06,
00308
00309
             1.009e-05, 1.119e-05, 1.228e-05, 1.365e-05, 1.506e-05, 1.641e-05,
00310
              1.784e-05, 1.952e-05, 2.132e-05, 2.323e-05, 2.531e-05, 2.754e-05,
00311
00312
              3.047e-05, 3.459e-05, 3.922e-05, 4.439e-05, 4.825e-05, 5.077e-05,
00313
              5.34e-05, 5.618e-05, 5.909e-05, 6.207e-05, 6.519e-05, 6.845e-05,
             6.819e-05, 6.726e-05, 6.622e-05, 6.512e-05, 6.671e-05, 6.862e-05, 7.048e-05, 7.264e-05, 7.3e-05, 7.3e-05, 7.3e-05, 7.3e-05, 7.3e-05
00314
00315
00316
00318
           static double cof2[121] = {
00319
             7.5e-14, 1.055e-13, 1.485e-13, 2.111e-13, 3.001e-13, 4.333e-13,
             6.269e-13, 9.221e-13, 1.364e-12, 2.046e-12, 3.093e-12, 4.703e-12, 7.225e-12, 1.113e-11, 1.66e-11, 2.088e-11, 2.626e-11, 3.433e-11, 4.549e-11, 5.886e-11, 7.21e-11, 8.824e-11, 1.015e-10, 1.155e-10,
00320
00321
00322
```

```
1.288e-10, 1.388e-10, 1.497e-10, 1.554e-10, 1.606e-10, 1.639e-10,
               1.64e-10, 1.64e-10, 1.596e-10, 1.542e-10, 1.482e-10, 1.382e-10,
00324
00325
               1.289e-10, 1.198e-10, 1.109e-10, 1.026e-10, 9.484e-11, 8.75e-11,
               8.086e-11, 7.49e-11, 6.948e-11, 6.446e-11, 5.961e-11, 5.505e-11, 5.085e-11, 4.586e-11, 4.1e-11, 3.665e-11, 3.235e-11, 2.842e-11, 2.491e-11, 2.11e-11, 1.769e-11, 1.479e-11, 1.197e-11, 9.631e-12,
00326
00327
00328
               7.74e-12, 6.201e-12, 4.963e-12, 3.956e-12, 3.151e-12, 2.507e-12, 1.99e-12, 1.576e-12, 1.245e-12, 9.83e-13, 7.742e-13, 6.088e-13,
00330
00331
               4.782e-13, 3.745e-13, 2.929e-13, 2.286e-13, 1.782e-13, 1.388e-13,
00332
               1.079 e^{-13},\ 8.362 e^{-14},\ 6.471 e^{-14},\ 4.996 e^{-14},\ 3.85 e^{-14},\ 2.96 e^{-14},
               2.265e-14, 1.729e-14, 1.317e-14, 9.998e-15, 7.549e-15, 5.683e-15,
00333
00334
               4.273e-15, 3.193e-15, 2.385e-15, 1.782e-15, 1.331e-15, 9.957e-16,
00335
                7.461e-16, 5.601e-16, 4.228e-16, 3.201e-16, 2.438e-16, 1.878e-16,
               1.445e-16, 1.111e-16, 8.544e-17, 6.734e-17, 5.341e-17, 4.237e-17,
00336
00337
               3.394e-17, 2.759e-17, 2.254e-17, 1.851e-17, 1.54e-17, 1.297e-17,
00338
               1.096e-17, 9.365e-18, 8e-18, 6.938e-18, 6.056e-18, 5.287e-18,
00339
               4.662e-18
00340
00341
00342
            static double f11[121] = {
               2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10,
00343
00344
               2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.635e-10, 2.536e-10,
               00345
00346
00347
00348
               5.993e-12, 3.345e-12, 1.839e-12, 9.264e-13, 4.688e-13, 2.329e-13,
00349
               1.129e-13, 5.505e-14, 2.825e-14, 1.492e-14, 7.997e-15, 5.384e-15,
00350
               3.988e-15, 2.955e-15, 2.196e-15, 1.632e-15, 1.214e-15, 9.025e-16,
00351
               6.708e-16, 4.984e-16, 3.693e-16, 2.733e-16, 2.013e-16, 1.481e-16,
               1.087e-16, 7.945e-17, 5.782e-17, 4.195e-17, 3.038e-17, 2.19e-17,
00352
               1.577e-17, 1.128e-17, 8.063e-18, 5.753e-18, 4.09e-18, 2.899e-18,
00353
00354
               2.048e-18, 1.444e-18, 1.015e-18, 7.12e-19, 4.985e-19, 3.474e-19,
00355
               2.417e-19, 1.677e-19, 1.161e-19, 8.029e-20, 5.533e-20, 3.799e-20,
00356
               2.602e-20, 1.776e-20, 1.209e-20, 8.202e-21, 5.522e-21, 3.707e-21,
00357
               2.48e-21, 1.652e-21, 1.091e-21, 7.174e-22, 4.709e-22, 3.063e-22,
               1.991e-22, 1.294e-22, 8.412e-23, 5.483e-23, 3.581e-23, 2.345e-23,
00358
00359
               1.548e-23, 1.027e-23, 6.869e-24, 4.673e-24, 3.173e-24, 2.153e-24,
00360
                1.461e-24, 1.028e-24, 7.302e-25, 5.188e-25, 3.739e-25, 2.753e-25,
00361
               2.043e-25, 1.528e-25, 1.164e-25, 9.041e-26, 7.051e-26, 5.587e-26,
00362
               4.428e-26, 3.588e-26, 2.936e-26, 2.402e-26, 1.995e-26
00363
00364
            static double f12[121] = {
00365
00366
               5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10,
               5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.45e-10, 5.429e-10, 5.291e-10,
00367
00368
               5.155e-10, 5.022e-10, 4.893e-10, 4.772e-10, 4.655e-10, 4.497e-10,
00369
               4.249e-10, 4.015e-10, 3.632e-10, 3.261e-10, 2.858e-10, 2.408e-10,
               2.03e-10, 1.685e-10, 1.4e-10, 1.163e-10, 9.65e-11, 8.02e-11, 6.705e-11, 5.624e-11, 4.764e-11, 4.249e-11, 3.792e-11, 3.315e-11, 2.819e-11, 2.4e-11, 1.999e-11, 1.64e-11, 1.352e-11, 1.14e-11, 9.714e-12,
00370
00371
00372
               8.28e-12, 7.176e-12, 6.251e-12, 5.446e-12, 4.72e-12, 4.081e-12,
00374
               3.528e-12, 3.08e-12, 2.699e-12, 2.359e-12, 2.111e-12, 1.901e-12,
00375
               1.709e-12, 1.534e-12, 1.376e-12, 1.233e-12, 1.103e-12, 9.869e-13,
00376
               8.808e-13, 7.859e-13, 7.008e-13, 6.241e-13, 5.553e-13, 4.935e-13,
               4.383e-13, 3.889e-13, 3.447e-13, 3.054e-13, 2.702e-13, 2.389e-13,
00377
               2.11e-13, 1.862e-13, 1.643e-13, 1.448e-13, 1.274e-13, 1.121e-13, 9.844e-14, 8.638e-14, 7.572e-14, 6.62e-14, 5.782e-14, 5.045e-14,
00378
               4.394e-14, 3.817e-14, 3.311e-14, 2.87e-14, 2.48e-14, 2.142e-14,
00380
00381
               1.851e-14, 1.599e-14, 1.383e-14, 1.196e-14, 1.036e-14, 9e-15,
               7.828e-15, 6.829e-15, 5.992e-15, 5.254e-15, 4.606e-15, 4.037e-15, 3.583e-15, 3.19e-15, 2.841e-15, 2.542e-15, 2.291e-15, 2.07e-15, 1.875e-15, 1.71e-15, 1.57e-15, 1.442e-15, 1.333e-15, 1.232e-15,
00382
00383
00384
00385
               1.147e-15, 1.071e-15, 1.001e-15, 9.396e-16
00386
00387
00388
            static double f14[121] = {
              9e-11, 8.73e-11, 8.46e-11, 8.19e-11, 7.92e-11, 7.74e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00389
00390
00391
00392
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00393
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00394
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00395
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00396
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                             7.65e-11,
00397
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00398
00399
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                             7.65e-11,
                                                                                                             7.65e-11,
00400
               7.65e-11, 7.65e-11, 7.65e-11,
                                                              7.65e-11, 7.65e-11,
                                                                                             7.65e-11,
                                                                                              7.65e-11,
                                                                                                             7.65e-11,
00401
               7.65e-11, 7.65e-11,
                                               7.65e-11,
                                                              7.65e-11, 7.65e-11,
00402
                7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-
00403
00404
               7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11
00405
00406
00407
00408
            static double f22[121] = {
00409
               1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10,
```

```
1.4e-10, 1.4e-10, 1.4e-10, 1.372e-10, 1.317e-10, 1.235e-10, 1.153e-10,
           1.075e-10, 1.002e-10, 9.332e-11, 8.738e-11, 8.194e-11, 7.7e-11, 7.165e-11, 6.753e-11, 6.341e-11, 5.971e-11, 5.6e-11, 5.229e-11,
00411
00412
           4.859e-11, 4.488e-11, 4.118e-11, 3.83e-11, 3.568e-11, 3.308e-11, 3.047e-11, 2.82e-11, 2.594e-11, 2.409e-11, 2.237e-11, 2.065e-11, 1.894e-11, 1.771e-11, 1.647e-11, 1.532e-11, 1.416e-11, 1.332e-11,
00413
00414
00415
            1.246e-11, 1.161e-11, 1.087e-11, 1.017e-11, 9.471e-12, 8.853e-12,
            8.235e-12, 7.741e-12, 7.247e-12, 6.836e-12, 6.506e-12, 6.176e-12,
00417
00418
           5.913e-12, 5.65e-12, 5.419e-12, 5.221e-12, 5.024e-12, 4.859e-12,
           3.31e-12, 3.54e-12, 4.414e-12, 4.282e-12, 4.15e-12, 4.019e-12, 3.903e-12, 3.805e-12, 3.706e-12, 3.607e-12, 3.508e-12, 3.41e-12, 3.31e-12, 3.212e-12, 3.129e-12, 3.047e-12, 2.964e-12, 2.882e-12, 2.8e-12, 2.734e-12, 2.668e-12, 2.602e-12, 2.537e-12, 2.471e-12,
00419
00420
00421
00422
           2.421e-12, 2.372e-12, 2.322e-12, 2.273e-12, 2.224e-12, 2.182e-12,
00423
00424
           2.141e-12, 2.1e-12, 2.059e-12, 2.018e-12, 1.977e-12, 1.935e-12,
00425
           1.894e-12, 1.853e-12, 1.812e-12, 1.77e-12, 1.73e-12, 1.688e-12,
           1.647e-12, 1.606e-12, 1.565e-12, 1.524e-12, 1.483e-12, 1.441e-12,
00426
           1.4e-12, 1.359e-12, 1.317e-12, 1.276e-12, 1.235e-12, 1.194e-12, 1.153e-12, 1.112e-12, 1.071e-12, 1.029e-12, 9.883e-13
00427
00429
00430
00431
         static double h2o[121] = {
           0.01166, 0.008269, 0.005742, 0.003845, 0.00277, 0.001897, 0.001272, 0.000827, 0.000539, 0.0003469, 0.0001579, 3.134e-05, 1.341e-05,
00432
00433
00434
           6.764e-06, 4.498e-06, 3.703e-06, 3.724e-06, 3.899e-06, 4.002e-06,
            4.122e-06, 4.277e-06, 4.438e-06, 4.558e-06, 4.673e-06, 4.763e-06,
            4.809e-06, 4.856e-06, 4.936e-06, 5.021e-06, 5.114e-06, 5.222e-06,
00436
00437
           5.331e-06, 5.414e-06, 5.488e-06, 5.563e-06, 5.633e-06, 5.704e-06,
00438
           5.767e-06, 5.819e-06, 5.872e-06, 5.914e-06, 5.949e-06, 5.984e-06,
00439
           6.015e-06, 6.044e-06, 6.073e-06, 6.104e-06, 6.136e-06, 6.167e-06,
00440
           6.189e-06, 6.208e-06, 6.226e-06, 6.212e-06, 6.185e-06, 6.158e-06,
00441
            6.114e-06, 6.066e-06, 6.018e-06, 5.877e-06, 5.728e-06, 5.582e-06,
           5.437e-06, 5.296e-06, 5.156e-06, 5.02e-06, 4.886e-06, 4.754e-06,
00442
00443
            4.625e-06, 4.498e-06, 4.374e-06, 4.242e-06, 4.096e-06, 3.955e-06,
           3.817e-06, 3.683e-06, 3.491e-06, 3.204e-06, 2.94e-06, 2.696e-06, 2.47e-06, 2.252e-06, 2.019e-06, 1.808e-06, 1.618e-06, 1.445e-06,
00444
00445
            1.285e-06, 1.105e-06, 9.489e-07, 8.121e-07, 6.938e-07, 5.924e-07,
00446
            5.04e-07, 4.288e-07, 3.648e-07, 3.103e-07, 2.642e-07, 2.252e-07,
00448
            1.921e-07, 1.643e-07, 1.408e-07, 1.211e-07, 1.048e-07, 9.063e-08,
00449
            7.835e-08, 6.774e-08, 5.936e-08, 5.221e-08, 4.592e-08, 4.061e-08,
00450
           3.62e-08, 3.236e-08, 2.902e-08, 2.62e-08, 2.383e-08, 2.171e-08,
           1.989e-08, 1.823e-08, 1.684e-08, 1.562e-08, 1.449e-08, 1.351e-08
00451
00452
00453
         static double h2o2[121] =
00455
            1.779e-10, 7.938e-10, 8.953e-10, 8.032e-10, 6.564e-10, 5.159e-10,
00456
           4.003e-10, 3.026e-10, 2.222e-10, 1.58e-10, 1.044e-10, 6.605e-11,
00457
           3.413e-11, 1.453e-11, 1.062e-11, 1.009e-11, 9.597e-12, 1.175e-11,
           1.572e-11, 2.091e-11, 2.746e-11, 3.603e-11, 4.791e-11, 6.387e-11, 8.239e-11, 1.007e-10, 1.23e-10, 1.363e-10, 1.489e-10, 1.585e-10,
00458
00459
           1.608e-10, 1.632e-10, 1.576e-10, 1.502e-10, 1.423e-10, 1.302e-10,
            1.192e-10, 1.085e-10, 9.795e-11, 8.854e-11, 8.057e-11, 7.36e-11,
00461
00462
            6.736e-11, 6.362e-11, 6.087e-11, 5.825e-11, 5.623e-11, 5.443e-11,
00463
           5.27 e-11, \ 5.098 e-11, \ 4.931 e-11, \ 4.769 e-11, \ 4.611 e-11, \ 4.458 e-11,
00464
           4.308e-11, 4.102e-11, 3.887e-11, 3.682e-11, 3.521e-11, 3.369e-11,
            3.224e-11, 3.082e-11, 2.946e-11, 2.814e-11, 2.687e-11, 2.566e-11,
00465
            2.449e-11, 2.336e-11, 2.227e-11, 2.123e-11, 2.023e-11, 1.927e-11,
           1.835e-11, 1.746e-11, 1.661e-11, 1.58e-11, 1.502e-11, 1.428e-11,
00467
00468
            1.357e-11, 1.289e-11, 1.224e-11, 1.161e-11, 1.102e-11, 1.045e-11,
00469
            9.895e-12, 9.369e-12, 8.866e-12, 8.386e-12, 7.922e-12, 7.479e-12,
           7.06e-12, 6.656e-12, 6.274e-12, 5.914e-12, 5.575e-12, 5.257e-12, 4.959e-12, 4.679e-12, 4.42e-12, 4.178e-12, 3.954e-12, 3.75e-12,
00470
00471
00472
           3.557e-12, 3.372e-12, 3.198e-12, 3.047e-12, 2.908e-12, 2.775e-12,
            2.653e-12, 2.544e-12, 2.442e-12, 2.346e-12, 2.26e-12, 2.183e-12,
00473
00474
           2.11e-12, 2.044e-12, 1.98e-12, 1.924e-12, 1.871e-12, 1.821e-12,
00475
           1.775e-12
00476
00477
00478
         static double hcn[121] = {
           5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10,
            5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.498e-10, 5.495e-10, 5.493e-10,
00480
00481
           5.49e-10, 5.488e-10, 4.717e-10, 3.946e-10, 3.174e-10, 2.4e-10,
           1.626e-10, 1.619e-10, 1.612e-10, 1.602e-10, 1.593e-10, 1.582e-10, 1.572e-10, 1.56e-10, 1.549e-10, 1.539e-10, 1.53e-10, 1.519e-10,
00482
00483
           1.506e-10, 1.487e-10, 1.467e-10, 1.449e-10, 1.43e-10, 1.413e-10,
00484
            1.397e-10, 1.382e-10, 1.368e-10, 1.354e-10, 1.337e-10, 1.315e-10,
           1.292e-10, 1.267e-10, 1.241e-10, 1.215e-10, 1.19e-10, 1.165e-10,
00486
00487
            1.141e-10, 1.118e-10, 1.096e-10, 1.072e-10, 1.047e-10, 1.021e-10,
00488
            9.968e-11, 9.739e-11, 9.539e-11, 9.339e-11, 9.135e-11, 8.898e-11,
           8.664e-11, 8.439e-11, 8.249e-11, 8.075e-11, 7.904e-11, 7.735e-11,
00489
00490
            7.565e-11, 7.399e-11, 7.245e-11, 7.109e-11, 6.982e-11, 6.863e-11,
            6.755e-11, 6.657e-11, 6.587e-11, 6.527e-11, 6.476e-11, 6.428e-11,
            6.382e-11, 6.343e-11, 6.307e-11, 6.272e-11, 6.238e-11, 6.205e-11,
00492
00493
            6.17e-11, 6.137e-11, 6.102e-11, 6.072e-11, 6.046e-11, 6.03e-11,
00494
            6.018e-11, 6.01e-11, 6.001e-11, 5.992e-11, 5.984e-11, 5.975e-11,
           5.967e-11, 5.958e-11, 5.95e-11, 5.941e-11, 5.933e-11, 5.925e-11, 5.916e-11, 5.908e-11, 5.899e-11, 5.891e-11, 5.883e-11, 5.874e-11,
00495
00496
```

```
5.866e-11, 5.858e-11, 5.85e-11, 5.841e-11, 5.833e-11, 5.825e-11,
            5.817e-11, 5.808e-11, 5.8e-11, 5.792e-11, 5.784e-11
00498
00499
00500
00501
          static double hno3[121] = {
            1.809e-10, 7.234e-10, 5.899e-10, 4.342e-10, 3.277e-10, 2.661e-10,
00502
            2.35e-10, 2.267e-10, 2.389e-10, 2.651e-10, 3.255e-10, 4.099e-10,
            5.42e-10, 6.978e-10, 8.807e-10, 1.112e-09, 1.405e-09, 2.04e-09,
00504
00505
            3.111e-09, 4.5e-09, 5.762e-09, 7.37e-09, 7.852e-09, 8.109e-09,
00506
            8.067e-09, 7.554e-09, 7.076e-09, 6.268e-09, 5.524e-09, 4.749e-09,
            3.909e-09, 3.223e-09, 2.517e-09, 1.942e-09, 1.493e-09, 1.122e-09,
00507
00508
            8.449e-10, 6.361e-10, 4.787e-10, 3.611e-10, 2.804e-10, 2.215e-10,
            1.758e-10, 1.441e-10, 1.197e-10, 9.953e-11, 8.505e-11, 7.334e-11,
            6.325e-11, 5.625e-11, 5.058e-11, 4.548e-11, 4.122e-11, 3.748e-11,
00510
00511
            3.402e-11, 3.088e-11, 2.8e-11, 2.536e-11, 2.293e-11, 2.072e-11,
00512
            1.871e-11, 1.687e-11, 1.52e-11, 1.368e-11, 1.23e-11, 1.105e-11,
            9.922e-12, 8.898e-12, 7.972e-12, 7.139e-12, 6.385e-12, 5.708e-12,
00513
            5.099e-12, 4.549e-12, 4.056e-12, 3.613e-12, 3.216e-12, 2.862e-12,
00514
            2.544e-12, 2.259e-12, 2.004e-12, 1.776e-12, 1.572e-12, 1.391e-12,
00516
            1.227e-12, 1.082e-12, 9.528e-13, 8.379e-13, 7.349e-13, 6.436e-13,
00517
            5.634e-13, 4.917e-13, 4.291e-13, 3.745e-13, 3.267e-13, 2.854e-13,
00518
            2.494e-13, 2.181e-13, 1.913e-13, 1.68e-13, 1.479e-13, 1.31e-13,
            1.159e-13, 1.025e-13, 9.067e-14, 8.113e-14, 7.281e-14, 6.535e-14, 5.892e-14, 5.348e-14, 4.867e-14, 4.439e-14, 4.073e-14, 3.76e-14, 3.476e-14, 3.229e-14, 3e-14, 2.807e-14, 2.635e-14, 2.473e-14,
00519
00520
00521
00522
00523
00524
00525
          static double hno4[121] = {
            6.118e-12, 3.594e-12, 2.807e-12, 3.04e-12, 4.458e-12, 7.986e-12,
00526
            1.509e-11, 2.661e-11, 3.738e-11, 4.652e-11, 4.429e-11, 3.992e-11, 3.347e-11, 3.005e-11, 3.173e-11, 4.055e-11, 5.812e-11, 8.489e-11,
00527
            1.19e-10, 1.482e-10, 1.766e-10, 2.103e-10, 2.35e-10, 2.598e-10,
00529
00530
            2.801e-10, 2.899e-10, 3e-10, 2.817e-10, 2.617e-10, 2.332e-10,
            1.933e-10, 1.605e-10, 1.232e-10, 9.285e-11, 6.941e-11, 4.951e-11, 3.539e-11, 2.402e-11, 1.522e-11, 9.676e-12, 6.056e-12, 3.745e-12, 2.34e-12, 1.463e-12, 9.186e-13, 5.769e-13, 3.322e-13, 1.853e-13, 1.035e-13, 7.173e-14, 5.382e-14, 4.036e-14, 3.401e-14, 2.997e-14,
00531
00532
00533
00535
            2.635e-14, 2.316e-14, 2.034e-14, 1.783e-14, 1.56e-14, 1.363e-14,
00536
            1.19e-14, 1.037e-14, 9.032e-15, 7.846e-15, 6.813e-15, 5.912e-15,
00537
            5.121e-15, 4.431e-15, 3.829e-15, 3.306e-15, 2.851e-15, 2.456e-15,
            2.114e-15, 1.816e-15, 1.559e-15, 1.337e-15, 1.146e-15, 9.811e-16,
00538
            8.389e-16, 7.162e-16, 6.109e-16, 5.203e-16, 4.425e-16, 3.76e-16, 3.184e-16, 2.692e-16, 2.274e-16, 1.917e-16, 1.61e-16, 1.35e-16,
00539
            1.131e-16, 9.437e-17, 7.874e-17, 6.57e-17, 5.481e-17, 4.579e-17, 3.828e-17, 3.204e-17, 2.691e-17, 2.264e-17, 1.912e-17, 1.626e-17,
00541
00542
00543
            1.382e-17, 1.174e-17, 9.972e-18, 8.603e-18, 7.45e-18, 6.453e-18,
            5.623e-18, 4.944e-18, 4.361e-18, 3.859e-18, 3.443e-18, 3.096e-18, 2.788e-18, 2.528e-18, 2.293e-18, 2.099e-18, 1.929e-18, 1.773e-18,
00544
00545
            1.64e-18
00546
00548
00549
          static double hocl[121] = {
           1.056e-12, 1.194e-12, 1.35e-12, 1.531e-12, 1.737e-12, 1.982e-12, 2.263e-12, 2.599e-12, 2.991e-12, 3.459e-12, 4.012e-12, 4.662e-12, 5.438e-12, 6.35e-12, 7.425e-12, 8.686e-12, 1.016e-11, 1.188e-11,
00550
00551
00552
            1.389e-11, 1.659e-11, 2.087e-11, 2.621e-11, 3.265e-11, 4.064e-11,
            4.859e-11, 5.441e-11, 6.09e-11, 6.373e-11, 6.611e-11, 6.94e-11,
00554
00555
            7.44e-11, 7.97e-11, 8.775e-11, 9.722e-11, 1.064e-10, 1.089e-10,
00556
            1.114e-10, 1.106e-10, 1.053e-10, 1.004e-10, 9.006e-11, 7.778e-11,
00557
            6.739e-11, 5.636e-11, 4.655e-11, 3.845e-11, 3.042e-11, 2.368e-11,
00558
            1.845e-11, 1.442e-11, 1.127e-11, 8.814e-12, 6.544e-12, 4.763e-12,
            3.449e-12, 2.612e-12, 1.999e-12, 1.526e-12, 1.16e-12, 8.793e-13,
            6.655e-13, 5.017e-13, 3.778e-13, 2.829e-13, 2.117e-13, 1.582e-13,
00560
00561
            1.178e-13, 8.755e-14, 6.486e-14,
                                                      4.799e-14, 3.54e-14, 2.606e-14,
00562
            1.916e-14, 1.403e-14, 1.026e-14, 7.48e-15, 5.446e-15, 3.961e-15,
00563
            2.872e-15, 2.076e-15, 1.498e-15, 1.077e-15, 7.726e-16, 5.528e-16,
            3.929e-16, 2.785e-16, 1.969e-16, 1.386e-16, 9.69e-17, 6.747e-17,
00564
00565
            4.692e-17, 3.236e-17, 2.232e-17, 1.539e-17, 1.061e-17, 7.332e-18,
            5.076e-18, 3.522e-18, 2.461e-18, 1.726e-18, 1.22e-18, 8.75e-19,
            6.264e-19, 4.482e-19, 3.207e-19, 2.368e-19, 1.762e-19, 1.312e-19, 9.891e-20, 7.595e-20, 5.87e-20, 4.567e-20, 3.612e-20, 2.904e-20,
00567
00568
00569
            2.343e-20, 1.917e-20, 1.568e-20, 1.308e-20, 1.1e-20, 9.25e-21,
00570
            7.881e-21
00571
00573
          static double n2o[121] = {
          3.17e-07, 3.03e-07, 2.984e-07, 2.938e-07, 2.892e-07, 2.847e-07, 2.779e-07, 2.705e-07, 2.631e-07, 2.557e-07, 2.484e-07, 2.345e-07, 2.201e-07, 2.01e-07,
00574
00575
00576
00577
             1.754e-07, 1.532e-07, 1.329e-07, 1.154e-07, 1.003e-07, 8.735e-08,
            7.617e-08, 6.512e-08, 5.547e-08, 4.709e-08, 3.915e-08, 3.259e-08,
00579
00580
            2.738e-08, 2.327e-08, 1.98e-08, 1.711e-08, 1.493e-08, 1.306e-08,
00581
            1.165e-08, 1.049e-08, 9.439e-09, 8.375e-09, 7.391e-09, 6.525e-09,
            5.759e-09, 5.083e-09, 4.485e-09, 3.953e-09, 3.601e-09, 3.27e-09, 2.975e-09, 2.757e-09, 2.556e-09, 2.37e-09, 2.195e-09, 2.032e-09,
00582
00583
```

```
1.912e-09, 1.79e-09, 1.679e-09, 1.572e-09, 1.482e-09, 1.402e-09,
                    1.326e-09, 1.254e-09, 1.187e-09, 1.127e-09, 1.071e-09, 1.02e-09,
00585
00586
                    9.673e-10, 9.193e-10, 8.752e-10, 8.379e-10, 8.017e-10, 7.66e-10,
00587
                   7.319e-10, 7.004e-10, 6.721e-10, 6.459e-10, 6.199e-10, 5.942e-10,
                   5.703e-10, 5.488e-10, 5.283e-10, 5.082e-10, 4.877e-10, 4.696e-10, 4.52e-10, 4.355e-10, 4.198e-10, 4.039e-10, 3.888e-10, 3.754e-10,
00588
00589
                   3.624e-10, 3.499e-10, 3.381e-10, 3.267e-10, 3.163e-10, 3.058e-10, 2.959e-10, 2.864e-10, 2.77e-10, 2.686e-10, 2.604e-10, 2.534e-10,
00591
00592
                   2.462e-10, 2.386e-10, 2.318e-10, 2.247e-10, 2.189e-10, 2.133e-10,
00593
                   2.071e-10, 2.014e-10, 1.955e-10, 1.908e-10, 1.86e-10, 1.817e-10
00594
00595
00596
               static double n2o5[121] = +
                 1.231e-11, 3.035e-12, 1.702e-12, 9.877e-13, 8.081e-13, 9.039e-13,
00597
00598
                   1.169e-12, 1.474e-12, 1.651e-12, 1.795e-12, 1.998e-12, 2.543e-12,
00599
                    4.398e-12, 7.698e-12, 1.28e-11, 2.131e-11, 3.548e-11, 5.894e-11,
00600
                    7.645e-11, 1.089e-10, 1.391e-10, 1.886e-10, 2.386e-10, 2.986e-10,
                   3.487e-10, 3.994e-10, 4.5e-10, 4.6e-10, 4.591e-10, 4.1e-10, 3.488e-10,
00601
                   2.846e-10, 2.287e-10, 1.696e-10, 1.011e-10, 6.428e-11, 4.324e-11,
                    2.225e-11, 6.214e-12, 3.608e-12, 8.793e-13, 4.491e-13, 1.04e-13,
                    6.1e-14, 3.436e-14, 6.671e-15, 1.171e-15, 5.848e-16, 1.212e-16,
00604
00605
                   le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00606
                   1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
                   le-16, le
00607
00608
                    le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00610
                    le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00611
                   1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00612
                   1e-16, 1e-16
00613
00614
00615
               static double nh3[121] = {
                  1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10,
00616
00617
                   1e-10, 1e-10, 1e-10, 1e-10, 9.444e-11, 8.488e-11, 7.241e-11, 5.785e-11,
                   4.178e-11, 3.018e-11, 2.18e-11, 1.574e-11, 1.137e-11, 8.211e-12, 5.973e-12, 4.327e-12, 3.118e-12, 2.234e-12, 1.573e-12, 1.04e-12,
00618
00619
                   6.762e-13, 4.202e-13, 2.406e-13, 1.335e-13, 6.938e-14, 3.105e-14,
00620
                   1.609e-14, 1.033e-14, 6.432e-15, 4.031e-15, 2.555e-15, 1.656e-15,
00622
                    1.115e-15, 7.904e-16, 5.63e-16, 4.048e-16, 2.876e-16, 2.004e-16,
                   1.356=16, 9.237e=17, 6.235e=17, 4.223e=17, 3.009e=17, 2.328e=17, 2.002e=17, 1.914e=17, 1.914e=17, 1.914e=17, 1.914e=17, 1.914e=17,
00623
00624
                   1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00625
                   1.914e-17, 
00626
00627
                   1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00629
                   1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00630
                   1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00631
                   1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                    1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00632
                    1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00633
                    1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00635
                   1.914e-17
00636
00637
00638
               static double no[121] = {
00639
                  2.586e-10, 4.143e-11, 1.566e-11, 9.591e-12, 8.088e-12, 8.462e-12,
                    1.013e-11, 1.328e-11, 1.855e-11, 2.678e-11, 3.926e-11, 5.464e-11,
                    7.012e-11, 8.912e-11, 1.127e-10, 1.347e-10, 1.498e-10, 1.544e-10,
00641
00642
                   1.602e-10, 1.824e-10, 2.078e-10, 2.366e-10, 2.691e-10, 5.141e-10,
                   8.259e-10, 1.254e-09, 1.849e-09, 2.473e-09, 3.294e-09, 4.16e-09, 5.095e-09, 6.11e-09, 6.93e-09, 7.888e-09, 8.903e-09, 9.713e-09,
00643
00644
                   1.052e-08, 1.115e-08, 1.173e-08, 1.21e-08, 1.228e-08, 1.239e-08,
00645
00646
                    1.231e-08, 1.213e-08, 1.192e-08, 1.138e-08, 1.085e-08, 1.008e-08,
                    9.224e-09, 8.389e-09, 7.262e-09, 6.278e-09, 5.335e-09, 4.388e-09,
00648
                   3.589e-09, 2.761e-09, 2.129e-09, 1.633e-09, 1.243e-09, 9.681e-10,
00649
                   8.355e-10, 7.665e-10, 7.442e-10, 8.584e-10, 9.732e-10, 1.063e-09,
00650
                   1.163e-09, 1.286e-09, 1.472e-09, 1.707e-09, 2.032e-09, 2.474e-09,
                   2.977e-09, 3.506e-09, 4.102e-09, 5.013e-09, 6.493e-09, 8.414e-09,
00651
                   1.077e-08, 1.367e-08, 1.777e-08, 2.625e-08, 3.926e-08, 5.545e-08,
00652
                    7.195e-08, 9.464e-08, 1.404e-07, 2.183e-07, 3.329e-07, 4.535e-07,
                    6.158e-07, 8.187e-07, 1.075e-06, 1.422e-06, 1.979e-06, 2.71e-06,
00654
                   3.58e-06, 4.573e-06, 5.951e-06, 7.999e-06, 1.072e-05, 1.372e-05,
00655
                   1.697e-05, 2.112e-05, 2.643e-05, 3.288e-05, 3.994e-05, 4.794e-05, 5.606e-05, 6.383e-05, 7.286e-05, 8.156e-05, 8.883e-05, 9.469e-05, 9.848e-05, 0.0001023, 0.0001066, 0.0001115, 0.0001145, 0.0001142,
00656
00657
00658
                   0.0001133
00660
00661
00662
               static double no2[121] = {
                   3.036e-09, 2.945e-10, 9.982e-11, 5.069e-11, 3.485e-11, 2.982e-11,
00663
                   2.947e-11, 3.164e-11, 3.714e-11, 4.586e-11, 6.164e-11, 8.041e-11, 9.982e-11, 1.283e-10, 1.73e-10, 2.56e-10, 3.909e-10, 5.959e-10,
00664
                    9.081e-10, 1.384e-09, 1.788e-09, 2.189e-09, 2.686e-09, 3.091e-09,
00666
00667
                   3.49e-09, 3.796e-09, 4.2e-09, 5.103e-09, 6.005e-09, 6.3e-09, 6.706e-09,
00668
                   7.07e-09, 7.434e-09, 7.663e-09, 7.788e-09, 7.8e-09, 7.597e-09,
                   7.482e-09, 7.227e-09, 6.403e-09, 5.585e-09, 4.606e-09, 3.703e-09, 2.984e-09, 2.183e-09, 1.48e-09, 8.441e-10, 5.994e-10, 3.799e-10,
00669
00670
```

```
2.751e-10, 1.927e-10, 1.507e-10, 1.102e-10, 6.971e-11, 5.839e-11,
           3.904e-11, 3.087e-11, 2.176e-11, 1.464e-11, 1.209e-11, 8.497e-12, 6.477e-12, 4.371e-12, 2.914e-12, 2.424e-12, 1.753e-12, 1.35e-12,
00672
00673
00674
           9.417e-13, 6.622e-13, 5.148e-13, 3.841e-13, 3.446e-13, 3.01e-13,
           2.551e-13, 2.151e-13, 1.829e-13, 1.64e-13, 1.475e-13, 1.352e-13, 1.155e-13, 9.963e-14, 9.771e-14, 9.577e-14, 9.384e-14, 9.186e-14,
00675
00676
            9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
           9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
00678
00679
           9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
00680
           9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14
00681
00682
00683
         static double o3[121] = {
           2.218e-08, 3.394e-08, 3.869e-08, 4.219e-08, 4.501e-08, 4.778e-08,
00684
00685
           5.067e-08, 5.402e-08, 5.872e-08, 6.521e-08, 7.709e-08, 9.461e-08,
           1.269e-07, 1.853e-07, 2.723e-07, 3.964e-07, 5.773e-07, 8.2e-07, 1.155e-06, 1.59e-06, 2.076e-06, 2.706e-06, 3.249e-06, 3.848e-06,
00686
00687
           1.136 00, 1.136 00, 2.106 00, 2.106 00, 3.2476 00, 3.2476 00, 3.4476 00, 4.4596-06, 4.9866-06, 5.573e-06, 5.958e-06, 6.328e-06, 6.661e-06, 6.9e-06, 7.146e-06, 7.276e-06, 7.374e-06, 7.447e-06, 7.383e-06,
00688
            7.321e-06, 7.161e-06, 6.879e-06, 6.611e-06, 6.216e-06, 5.765e-06,
00690
           5.355e-06, 4.905e-06, 4.471e-06, 4.075e-06, 3.728e-06, 3.413e-06,
00691
00692
           3.125e-06, 2.856e-06, 2.607e-06, 2.379e-06, 2.17e-06, 1.978e-06,
           1.8e-06, 1.646e-06, 1.506e-06, 1.376e-06, 1.233e-06, 1.102e-06, 9.839e-07, 8.771e-07, 7.814e-07, 6.947e-07, 6.102e-07, 5.228e-07, 4.509e-07, 3.922e-07, 3.501e-07, 3.183e-07, 2.909e-07, 2.686e-07,
00693
00694
00695
           2.476e-07, 2.284e-07, 2.109e-07, 2.003e-07, 2.013e-07, 2.022e-07,
           2.032e-07, 2.042e-07, 2.097e-07, 2.361e-07, 2.656e-07,
00697
                                                                            2.989e-07.
           3.37e-07, 3.826e-07, 4.489e-07, 5.26e-07, 6.189e-07, 7.312e-07, 8.496e-07, 8.444e-07, 8.392e-07, 8.339e-07, 8.286e-07, 8.234e-07,
00698
00699
00700
           8.181e-07, 8.129e-07, 8.077e-07, 8.026e-07, 6.918e-07, 5.176e-07,
00701
           3.865e-07, 2.885e-07, 2.156e-07, 1.619e-07, 1.219e-07, 9.161e-08,
00702
           6.972e-08, 5.399e-08, 3.498e-08, 2.111e-08, 1.322e-08, 8.482e-09,
00703
           5.527e-09, 3.423e-09, 2.071e-09, 1.314e-09, 8.529e-10, 5.503e-10,
           3.665e-10
00704
00705
         };
00706
00707
         static double ocs[121] = {
           6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 5.997e-10,
00709
           5.989e-10, 5.881e-10, 5.765e-10, 5.433e-10, 5.074e-10, 4.567e-10,
00710
           4.067e-10, 3.601e-10, 3.093e-10, 2.619e-10, 2.232e-10,
                                                                            1.805e-10
00711
           1.46e-10, 1.187e-10, 8.03e-11, 5.435e-11, 3.686e-11, 2.217e-11,
           1.341e-11, 8.756e-12, 4.511e-12, 2.37e-12, 1.264e-12, 8.28e-13,
00712
           5.263e-13, 3.209e-13, 1.717e-13, 9.068e-14, 4.709e-14, 2.389e-14, 1.236e-14, 1.127e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00713
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00715
00716
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00717
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00718
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00719
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00720
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00722
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00723
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00724
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00725
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00726
           1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
           1.091e-14, 1.091e-14, 1.091e-14
00728
00729
00730
         static double sf6[121] = {
          4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12,
00731
           4.103e-12, 4.103e-12, 4.103e-12, 4.087e-12, 4.064e-12, 4.023e-12, 3.988e-12, 3.941e-12, 3.884e-12, 3.755e-12, 3.622e-12, 3.484e-12,
00732
           3.32e-12, 3.144e-12, 2.978e-12, 2.811e-12, 2.653e-12, 2.489e-12,
00734
00735
           2.332e-12, 2.199e-12, 2.089e-12, 2.013e-12, 1.953e-12, 1.898e-12,
00736
           1.859e-12, 1.826e-12, 1.798e-12, 1.776e-12, 1.757e-12, 1.742e-12,
00737
           1.728e-12, 1.717e-12, 1.707e-12, 1.698e-12, 1.691e-12, 1.685e-12,
00738
           1.679e-12, 1.675e-12, 1.671e-12, 1.668e-12, 1.665e-12, 1.663e-12,
           1.661e-12, 1.659e-12, 1.658e-12, 1.657e-12, 1.656e-12, 1.655e-12,
00740
           1.654e-12, 1.653e-12, 1.653e-12, 1.652e-12, 1.652e-12, 1.652e-12,
00741
           1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12,
                                                                            1.651e-12,
00742
           1.651e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00743
           1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00744
           1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00745
           1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00746
           1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00747
           1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00748
           1.65e-12, 1.65e-12
00749
00750
00751
         static double so2[121] = {
00753
           1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10, 1e-10,
00754
           1e-10, 1e-10, 9.867e-11, 9.537e-11, 9e-11, 8.404e-11, 7.799e-11,
00755
           7.205e-11, 6.616e-11, 6.036e-11, 5.475e-11, 5.007e-11, 4.638e-11,
00756
           4.346e-11, 4.055e-11, 3.763e-11, 3.471e-11, 3.186e-11, 2.905e-11, 2.631e-11, 2.358e-11, 2.415e-11, 2.949e-11, 3.952e-11, 5.155e-11,
00757
```

```
6.76e-11, 8.741e-11, 1.099e-10, 1.278e-10, 1.414e-10, 1.512e-10,
00759
                       1.607e-10, 1.699e-10, 1.774e-10, 1.832e-10, 1.871e-10, 1.907e-10, 1.943e-10, 1.974e-10, 1.993e-10, 2e-10, 2
00760
                       2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00761
00762
                       2e-10, 2e
00763
00764
                       2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00765
                       2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00766
                       2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00767
                       2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10
00768
00769
00770
                  static int ig co2 = -999;
00771
00772
                  double co2, \starq[NG] = { NULL };
00773
00774
                  int iq, ip, iw, iz;
00775
                 /* Find emitter index of CO2... */
00777
                  if (ig_co2 == -999)
00778
                      ig_co2 = find_emitter(ct1, "CO2");
00779
00780
                  /* Identify variable... */
00781
                  for (ig = 0; ig < ctl->ng; ig++) {
   q[iq] = NULL;
00782
00783
                       if (strcasecmp(ctl->emitter[ig], "C2H2") == 0)
                           q[ig] = c2h2;
00784
00785
                       if (strcasecmp(ctl->emitter[ig], "C2H6") == 0)
00786
                           q[ig] = c2h6;
00787
                       if (strcasecmp(ctl->emitter[iq], "CCl4") == 0)
00788
                           q[ig] = ccl4;
00789
                       if
                             (strcasecmp(ctl->emitter[ig], "CH4") == 0)
00790
                           q[ig] = ch4;
00791
                       if (strcasecmp(ctl->emitter[ig], "ClO") == 0)
00792
                            q[ig] = clo;
00793
                       if (strcasecmp(ctl->emitter[ig], "ClONO2") == 0)
00794
                          q[ig] = clono2;
                       if (strcasecmp(ctl->emitter[ig], "CO") == 0)
00796
                           q[ig] = co;
00797
                              (strcasecmp(ctl->emitter[ig], "COF2") == 0)
00798
                           q[ig] = cof2;
00799
                       if (strcasecmp(ctl->emitter[ig], "F11") == 0)
00800
                           q[ig] = f11;
00801
                       if (strcasecmp(ctl->emitter[ig], "F12") == 0)
00802
                           q[ig] = f12;
                              (strcasecmp(ctl->emitter[ig], "F14") == 0)
00803
                       if
00804
                           q[ig] = f14;
                       if (strcasecmp(ctl->emitter[ig], "F22") == 0)
00805
                           q[ig] = f22;
00806
                       if (strcasecmp(ctl->emitter[ig], "H2O") == 0)
00807
00808
                           q[ig] = h2o;
00809
                       if (strcasecmp(ctl->emitter[ig], "H2O2") == 0)
00810
                           q[ig] = h2o2;
00811
                       if (strcasecmp(ctl->emitter[ig], "HCN") == 0)
00812
                           q[ig] = hcn;
00813
                       if (strcasecmp(ctl->emitter[iq], "HNO3") == 0)
                           q[ig] = hno3;
00815
                              (strcasecmp(ctl->emitter[ig], "HNO4") == 0)
00816
                           q[ig] = hno4;
00817
                       if (strcasecmp(ctl->emitter[ig], "HOC1") == 0)
00818
                          q[ig] = hocl;
                       if (strcasecmp(ctl->emitter[ig], "N2O") == 0)
00819
00820
                           q[ig] = n20;
00821
                             (strcasecmp(ctl->emitter[ig], "N2O5") == 0)
00822
                           q[ig] = n2o5;
00823
                       if (strcasecmp(ctl->emitter[ig], "NH3") == 0)
00824
                           q[ig] = nh3;
00825
                       if (strcasecmp(ctl->emitter[ig], "NO") == 0)
                           q[ig] = no;
00826
                       if (strcasecmp(ctl->emitter[ig], "NO2") == 0)
00828
                           q[ig] = no2;
00829
                       if (strcasecmp(ctl->emitter[ig], "O3") == 0)
00830
                           q[ig] = o3;
                       if (strcasecmp(ctl->emitter[iq], "OCS") == 0)
00831
                           q[ig] = ocs;
00832
                        if (strcasecmp(ctl->emitter[ig], "SF6") == 0)
00833
00834
                           q[ig] = sf6;
00835
                              (strcasecmp(ctl->emitter[ig], "SO2") == 0)
00836
                           q[ig] = so2;
00837
00838
00839
                   /* Loop over atmospheric data points... */
                  for (ip = 0; ip < atm->np; ip++) {
00840
00841
00842
                        /\star Get altitude index... \star/
00843
                       iz = locate(z, 121, atm->z[ip]);
00844
```

```
/* Interpolate pressure... */
                      atm \rightarrow p[ip] = EXP(z[iz], pre[iz], z[iz + 1], pre[iz + 1], atm \rightarrow z[ip]);
00846
00847
00848
                       /* Interpolate temperature... */
                      atm \rightarrow t[ip] = LIN(z[iz], tem[iz], z[iz + 1], tem[iz + 1], atm \rightarrow z[ip]);
00849
00850
                       /* Interpolate trace gases... */
                       for (ig = 0; ig < ctl->ng; ig++)
00852
00853
                         if (q[ig] != NULL)
00854
                                atm->q[ig][ip]
                                   LIN(z[iz], q[ig][iz], z[iz + 1], q[ig][iz + 1], atm->z[ip]);
00855
00856
00857
                               atm->q[ig][ip] = 0;
00858
00859
                       /* Set CO2... */
00860
                       if (ig_co2 >= 0) {
00861
                           co2 =
                               371.789948e-6 + 2.026214e-6 * (atm->time[ip] - 63158400.) / 31557600.;
00862
00863
                           atm->q[ig\_co2][ip] = co2;
00864
00865
00866
                       /\star Set extinction to zero... \star/
00867
                      for (iw = 0; iw < ctl->nw; iw++)
00868
                          atm->k[iw][ip] = 0;
00869
00870 }
00871
00873
00874 double ctmco2(
00875
                 double nu.
00876
                 double p,
00877
                 double t
00878
                 double u)
00879
                  static double co2296[2001] = \{ 9.3388e-5, 9.7711e-5, 1.0224e-4, 1.0697e-4, 
00880
                     1.1193e-4, 1.1712e-4, 1.2255e-4, 1.2824e-4, 1.3419e-4, 1.4043e-4, 1.4695e-4, 1.5378e-4, 1.6094e-4, 1.6842e-4, 1.7626e-4, 1.8447e-4,
00881
00883
                       1.9307e-4, 2.0207e-4, 2.1149e-4, 2.2136e-4, 2.3169e-4, 2.4251e-4,
                      2.5384e-4, 2.657e-4, 2.7813e-4, 2.9114e-4, 3.0477e-4, 3.1904e-4,
00884
                      3.3399e-4, 3.4965e-4, 3.6604e-4, 3.8322e-4, 4.0121e-4, 4.2006e-4, 4.398e-4, 4.6047e-4, 4.8214e-4, 5.0483e-4, 5.286e-4, 5.535e-4,
00885
00886
                      5.7959e-4, 6.0693e-4, 6.3557e-4, 6.6558e-4, 6.9702e-4, 7.2996e-4, 7.6449e-4, 8.0066e-4, 8.3856e-4, 8.7829e-4, 9.1991e-4, 9.6354e-4,
00887
00888
                      .0010093, .0010572, .0011074, .00116, .0012152, .001273, .0013336, .0013972, .0014638, .0015336, .0016068, .0016835,
00890
00891
                       .001764, .0018483, .0019367, .0020295, .0021267, .0022286,
00892
                      .0023355, .0024476, .0025652, .0026885, .0028178, .0029534,
                      .0030956, .0032448, .0034012, .0035654, .0037375, .0039181, .0041076, .0043063, .0045148, .0047336, .0049632, .005204, .0054567, .0057219, .0060002, .0062923, .0065988, .0069204,
00893
00894
                      .007258, .0076123, .0079842, .0083746, .0087844, .0092146,
00896
00897
                      .0096663, .01014, .010638, .011161, .01171, .012286, .012891,
                      .013527, .014194, .014895, .015631, .016404, .017217, .01807, .018966, .019908, .020897, .021936, .023028, .024176, .025382, .026649, .027981, .02938, .030851, .032397, .034023, .035732, .037528, .039416, .041402, .04349, .045685, .047994, .050422, .052975, .055661, .058486, .061458, .064584, .067873, .071334,
00898
00899
00900
00902
00903
                      .074975, .078807, .082839, .087082, .091549, .096249, .1012,
00904
                       .10641, .11189, .11767, .12375, .13015, .13689, .14399, .15147,
                      .15031, .11603, .11707, .12573, .13013, .13003, .14339, .15147, .15935, .16765, .17639, .18561, .19531, .20554, .21632, .22769, .23967, .25229, .2656, .27964, .29443, .31004, .3265, .34386, .36218, .3815, .40188, .42339, .44609, .47004, .49533, .52202, .5502, .57995, .61137, .64455, .6796, .71663, .75574, .79707, .84075, .88691, .9357, .98728, 1.0418, 1.0995, 1.1605, 1.225,
00905
00906
00908
00909
                      1.2932, 1.3654, 1.4418, 1.5227, 1.6083, 1.6989, 1.7948, 1.8964, 2.004, 2.118, 2.2388, 2.3668, 2.5025, 2.6463, 2.7988, 2.9606, 3.1321, 3.314, 3.5071, 3.712, 3.9296, 4.1605, 4.4058, 4.6663, 4.9431, 5.2374, 5.5501, 5.8818, 6.2353, 6.6114, 7.0115, 7.4372,
00910
00911
00912
00913
                       7.8905, 8.3731, 8.8871, 9.4349, 10.019, 10.641, 11.305, 12.013,
00915
                       12.769, 13.576, 14.437, 15.358, 16.342, 17.39, 18.513, 19.716,
00916
                      21.003, 22.379, 23.854, 25.436, 27.126, 28.942, 30.89, 32.973,
                      35.219, 37.634, 40.224, 43.021, 46.037, 49.29, 52.803, 56.447, 60.418, 64.792, 69.526, 74.637, 80.182, 86.193, 92.713, 99.786, 107.47, 115.84, 124.94, 134.86, 145.69, 157.49, 170.3, 184.39, 199.83, 216.4, 234.55, 254.72, 276.82, 299.85, 326.16, 354.99,
00917
00918
00919
00920
                       386.51, 416.68, 449.89, 490.12, 534.35, 578.25, 632.26, 692.61,
00921
                      756.43, 834.75, 924.11, 1016.9, 996.96, 1102.7, 1219.2, 1351.9, 1494.3, 1654.1, 1826.5, 2027.9, 2249., 2453.8, 2714.4, 2999.4, 3209.5, 3509., 3840.4, 3907.5, 4190.7, 4533.5, 4648.3, 5059.1, 5561.6, 6191.4, 6820.8, 7905.9, 9362.2, 2431.3, 2211.3, 2046.8, 2023.8, 1985.9, 1905.9, 1491.1, 1369.8, 1262.2, 1200.7, 887.74,
00922
00923
00924
00925
                       820.25, 885.23, 887.21, 816.73, 1126.9, 1216.2, 1272.4, 1579.5,
00927
00928
                      1634.2, 1656.3, 1657.9, 1789.5, 1670.8, 1509.5, 8474.6, 7489.2
                      6793.6, 6117., 5574.1, 5141.2, 5084.6, 4745.1, 4413.2, 4102.8, 4024.7, 3715., 3398.6, 3100.8, 2900.4, 2629.2, 2374., 2144.7, 1955.8, 1760.8, 1591.2, 1435.2, 1296.2, 1174., 1065.1, 967.76,
00929
00930
00931
```

```
999.48, 897.45, 809.23, 732.77, 670.26, 611.93, 560.11, 518.77,
                  476.84, 438.8, 408.48, 380.21, 349.24, 322.71, 296.65, 272.85,
00933
00934
                  251.96, 232.04, 213.88, 197.69, 182.41, 168.41, 155.79, 144.05,
                  133.31, 123.48, 114.5, 106.21, 98.591, 91.612, 85.156, 79.204, 73.719, 68.666, 63.975, 59.637, 56.35, 52.545, 49.042, 45.788, 42.78, 39.992, 37.441, 35.037, 32.8, 30.744, 28.801, 26.986,
00935
00936
00937
                  25.297, 23.731, 22.258, 20.883, 19.603, 18.403, 17.295, 16.249
                  15.271, 14.356, 13.501, 12.701, 11.954, 11.254, 10.6, 9.9864,
00939
                  9.4118, 8.8745, 8.3714, 7.8997, 7.4578, 7.0446, 6.6573, 6.2949, 5.9577, 5.6395, 5.3419, 5.063, 4.8037, 4.5608, 4.3452, 4.1364, 3.9413, 3.7394, 3.562, 3.3932, 3.2325, 3.0789, 2.9318, 2.7898,
00940
00941
00942
                  2.6537, 2.5225, 2.3958, 2.2305, 2.1215, 2.0245, 1.3427, 1.8795, 1.8336, 1.7604, 1.7016, 1.6419, 1.5282, 1.4611, 1.3443, 1.27,
00943
00944
                  1.1675, 1.0824, 1.0534, .99833, .95854, .92981, .90887, .89346,
00945
00946
                  .88113, .87068, .86102, .85096, .88262, .86151, .83565, .80518,
00947
                  .77045, .73736, .74744, .74954, .75773, .82267, .83493, .89402,
                 89725, 93426, 95564, 94045, 94174, 93404, 92035, 90456, 88621, 86673, 78117, 7515, 72056, 68822, 65658, 62764, 55984, 55598, 57407, 60963, 63763, 66198, 61132, 60972, 52496, 50649, 41872, 33964, 32422, 27276, 24048, 23772,
00948
00949
00951
                  .2286, .22711, .23999, .32038, .34371, .36621, .38561, .39953, .40636, .44913, .42716, .3919, .35477, .33935, .3351, .39746, .40993, .49398, .49956, .56157, .54742, .57295, .57386, .55417,
00952
00953
00954
                  .50745, .471, .43446, .39102, .34993, .31269, .27888, .24912, .22291, .19994, .17972, .16197, .14633, .13252, .12029, .10942,
00955
00956
                  .099745, .091118, .083404, .076494, .070292, .064716, .059697, .055173, .051093, .047411, .044089, .041092, .038392, .035965,
00958
00959
                  .033789, .031846, .030122, .028607, .02729, .026169, .025209,
                  .024405, .023766, .023288, .022925, .022716, .022681, .022685,
00960
                 .022768, .023133, .023325, .023486, .024004, .024126, .024083, .023785, .024023, .023029, .021649, .021108, .019454, .017809, .017292, .016635, .017037, .018068, .018977, .018756, .017847, .016557, .016142, .014459, .012869, .012381, .010875, .0098701
00961
00962
00964
00965
                  .009285, .0091698, .0091701, .0096145, .010553, .01106, .012613,
                  .014362, .015017, .016507, .017741, .01768, .017784, .0171, .016357, .016172, .017257, .018978, .020935, .021741, .023567,
00966
00967
                  .025183, .025589, .026732, .027648, .028278, .028215, .02856, .029015, .029062, .028851, .028497, .027825, .027801, .026523, .02487, .022967, .022168, .020194, .018605, .017903, .018439,
00968
00970
                  .019697, .020311, .020855, .020057, .018608, .016738, .015963, .013844, .011801, .011134, .0097573, .0086007, .0086226, .0083721, .0090978, .0097616, .0098426, .011317, .012853, .014 .014657, .015771, .016351, .016079, .014829, .013431, .013185, .013207, .01448, .016176, .017971, .018265, .019526, .020455,
00971
00972
00973
00974
                 .019797, .019802, .0194, .018176, .017505, .016197, .015339, .015339, .014401, .013213, .012203, .011186, .010236, .0093288, .0084
00976
00977
00978
                  .0076837, .0069375, .0062614, .0056628, .0051153, .0046015,
                  .0041501, .003752, .0033996, .0030865, .0028077, .0025586, .0023355, .0021353, .0019553, .0017931, .0016466, .0015141, .0013941, .0012852, .0011862, .0010962, .0010142, 9.3935e-4, 8.71e-4, 8.0851e-4, 7.5132e-4, 6.9894e-4, 6.5093e-4, 6.0689e-4,
00979
00980
00981
                  5.6647e-4, 5.2935e-4, 4.9525e-4, 4.6391e-4, 4.3509e-4, 4.086e-4,
00983
00984
                  3.8424e-4, 3.6185e-4, 3.4126e-4, 3.2235e-4, 3.0498e-4, 2.8904e-4,
                  2.7444e-4, 2.6106e-4, 2.4883e-4, 2.3766e-4, 2.275e-4, 2.1827e-4, 2.0992e-4, 2.0239e-4, 1.9563e-4, 1.896e-4, 1.8427e-4, 1.796e-4, 1.7555e-4, 1.7209e-4, 1.692e-4, 1.6687e-4, 1.6505e-4, 1.6375e-4,
00985
00986
00987
                  1.6294e-4, 1.6261e-4, 1.6274e-4, 1.6334e-4, 1.6438e-4, 1.6587e-4,
                  1.678e-4, 1.7017e-4, 1.7297e-4, 1.762e-4, 1.7988e-4, 1.8399e-4,
00989
00990
                  1.8855e-4, 1.9355e-4, 1.9902e-4, 2.0494e-4, 2.1134e-4, 2.1823e-4,
                  2.2561e-4, 2.335e-4, 2.4192e-4, 2.5088e-4, 2.604e-4, 2.705e-4,
00991
                  2.8119e-4, 2.925le-4, 3.0447e-4, 3.171e-4, 3.3042e-4, 3.4447e-4, 3.5927e-4, 3.7486e-4, 3.9127e-4, 4.0854e-4, 4.267e-4, 4.4579e-4, 4.6586e-4, 4.8696e-4, 5.0912e-4, 5.324e-4, 5.5685e-4, 5.8253e-4,
00992
00993
                  6.0949e-4, 6.378e-4, 6.6753e-4, 6.9873e-4, 7.3149e-4, 7.6588e-4,
00995
00996
                  8.0198e-4, 8.3987e-4, 8.7964e-4, 9.2139e-4, 9.6522e-4,
                                                                                                                      .0010112
00997
                  .0010595, .0011102, .0011634, .0012193, .001278, .0013396,
                 .0014043, .0014722, .0015436, .0016185, .0016972, .0017799, .0018668, .001958, .0020539, .0021547, .0022606, .0023719, .002489, .002612, .0027414, .0028775, .0030206, .0031712, .0033295, .0034962, .0036716, .0038563, .0040506, .0042553,
00998
00999
01000
                  .0044709, .004698, .0049373, .0051894, .0054552, .0057354, .006031, .0063427, .0066717, .0070188, .0073854, .0077726,
01002
01003
                  .0081816, .0086138, .0090709, .0095543, .010066, .010607, .011181, .011789, .012433, .013116, .013842, .014613, .015432,
01004
01005
                  .016304, .017233, .018224, .019281, .020394, .021574, .022836, .024181, .025594, .027088, .028707, .030401, .032245, .034219, .036262, .038539, .040987, .043578, .04641, .04949, .052726,
01006
01008
                  .056326, .0602, .064093, .068521, .073278, .077734, .083064, .088731, .093885, .1003, .1072, .11365, .12187, .13078, .13989, .15095, .16299, .17634, .19116, .20628, .22419, .24386, .26587,
01009
01010
01011
                  .28811, .31399, .34321, .36606, .39675, .42742, .44243, .47197, .49993, .49027, .51147, .52803, .48931, .49729, .5026, .43854,
01012
                  .441, .44766, .43414, .46151, .50029, .55247, .43855, .32115, .32607, .3431, .36119, .38029, .41179, .43996, .47144, .51853
01014
01015
                  .55362, .59122, .66338, .69877, .74001, .82923, .86907, .90361,
01016
                  1.0025, 1.031, 1.0559, 1.104, 1.1178, 1.1341, 1.1547, 1.351, 1.4772, 1.4812, 1.4907, 1.512, 1.5442, 1.5853, 1.6358, 1.6963,
01017
```

```
01019
                         1.7674, 1.8474, 1.9353, 2.0335, 2.143, 2.2592, 2.3853, 2.5217,
                         2.6686, 2.8273, 2.9998, 3.183, 3.3688, 3.6109, 3.8564, 4.1159, 4.4079, 4.7278, 5.0497, 5.3695, 5.758, 6.0834, 6.4976, 6.9312,
01020
01021
                          7.38, 7.5746, 7.9833, 8.3791, 8.3956, 8.7501, 9.1067, 9.072,
01022
                         9.4649, 9.9112, 10.402, 10.829, 11.605, 12.54, 12.713, 10.443, 10.825, 11.375, 11.955, 12.623, 13.326, 14.101, 15.041, 15.547,
01023
01024
                         16.461, 17.439, 18.716, 19.84, 21.036, 22.642, 23.901, 25.244, 27.03, 28.411, 29.871, 31.403, 33.147, 34.744, 36.456, 39.239,
01026
                         43.605, 45.162, 47.004, 49.093, 51.391, 53.946, 56.673, 59.629, 63.167, 66.576, 70.254, 74.222, 78.477, 83.034, 87.914, 93.18, 98.77, 104.74, 111.15, 117.95, 125.23, 133.01, 141.33, 150.21, 159.71, 169.89, 180.93, 192.54, 204.99, 218.34, 232.65, 248., 264.47, 282.14, 301.13, 321.53, 343.48, 367.08, 392.5, 419.88,
01027
01028
01029
01030
                          449.4, 481.26, 515.64, 552.79, 592.99, 636.48, 683.61, 734.65,
01032
01033
                         789.99, 850.02, 915.14, 985.81, 1062.5, 1147.1, 1237.8, 1336.4,
                        1443.2, 1558.9, 1684.2, 1819.2, 1965.2, 2122.6, 2291.7, 2470.8, 2665.7, 2874.9, 3099.4, 3337.9, 3541., 3813.3, 4111.9, 4439.3, 4798.9, 5196., 5639.2, 6087.5, 6657.7, 7306.7, 8040.7, 8845.5, 9702.2, 10670., 11739., 12842., 14141., 15498., 17068., 18729., 20557., 22559., 25248., 27664., 30207., 32915., 35611., 38081.,
01034
01035
01036
01038
                          40715., 43191., 41651., 42750., 43785., 44353., 44366., 44189.,
01039
01040
                          43618., 42862., 41878., 35133., 35215., 36383., 39420., 44055.,
                          44155., 45850., 46853., 39197., 38274., 29942., 28553., 21792.,
01041
                         21228., 17106., 14955., 18181., 19557., 21427., 23728., 26301., 28584., 30775., 32536., 33867., 40089., 39204., 37329., 34452.,
01042
01043
                         31373., 33921., 34800., 36043., 44415., 45162., 52181., 50895., 54140., 50840., 50468., 48302., 44915., 40910., 36754., 32755.,
01045
                         29093., 25860., 22962., 20448., 18247., 16326., 14645., 13165., 11861., 10708., 9686.9, 8779.7, 7971.9, 7250.8, 6605.7, 6027.2, 5507.3, 5039.1, 4616.6, 4234.8, 3889., 3575.4, 3290.5, 3031.3, 2795.2, 2579.9, 2383.1, 2203.3, 2038.6, 1887.6, 1749.1, 1621.9,
01046
01047
01048
01049
                         1505., 1397.4, 1298.3, 1207., 1122.8, 1045., 973.1, 906.64, 845.16, 788.22, 735.48, 686.57, 641.21, 599.1, 559.99, 523.64, 489.85, 458.42, 429.16, 401.92, 376.54, 352.88, 330.82, 310.24,
01051
01052
                         291.03, 273.09, 256.34, 240.69, 226.05, 212.37, 199.57, 187.59, 176.37, 165.87, 156.03, 146.82, 138.17, 130.07, 122.47, 115.34, 108.65, 102.37, 96.473, 90.934, 85.73, 80.84, 76.243, 71.922, 67.858, 64.034, 60.438, 57.052, 53.866, 50.866, 48.04, 45.379, 42.872, 40.51, 38.285, 36.188, 34.211, 32.347, 30.588, 28.929,
01053
01054
01055
01057
                         27.362, 25.884, 24.489, 23.171, 21.929, 20.755, 19.646, 18.599, 17.61, 16.677, 15.795, 14.961, 14.174, 13.43, 12.725, 12.06,
01058
01059
                         11.431, 10.834, 10.27, 9.7361, 9.2302, 8.7518, 8.2997, 7.8724, 7.4674, 7.0848, 6.7226, 6.3794, 6.054, 5.745, 5.4525, 5.1752, 4.9121, 4.6625, 4.4259, 4.2015, 3.9888, 3.7872, 3.5961, 3.4149,
01060
01061
01062
                         3.2431, 3.0802, 2.9257, 2.7792, 2.6402, 2.5084, 2.3834, 2.2648, 2.1522, 2.0455, 1.9441, 1.848, 1.7567, 1.6701, 1.5878, 1.5097,
01063
01064
01065
                         1.4356, 1.3651, 1.2981, 1.2345, 1.174, 1.1167, 1.062, 1.0101,
                        01066
01067
01068
01070
                        .20081, .19143, .18261, .1/407, .16603, .15833, .15089, .14385, .13707, .13065, .12449, .11865, .11306, .10774, .10266, .097818, .093203, .088815, .084641, .080671, .076892, .073296, .069873, .066613, .06351, .060555, .05774, .055058, .052504, .050071, .047752, .045543, .043438, .041432, .039521, .037699, .035962, .034307, .032729, .031225, .029791, .028423, .02712, .025877, .024692, .023563, .022485, .021458, .020478, .019543, .018652,
01071
01072
01073
01074
01076
                        .017802, .016992, .016219, .015481, .014778, .014107, .013467, .012856, .012274, .011718, .011188, .010682, .0102, .0097393, .0093001, .008881, .0084812, .0080997, .0077358, .0073885, .0070571, .0067409, .0064393, .0061514, .0058768, .0056147, .0053647, .0051262, .0048987, .0046816, .0044745, .0042769,
01077
01078
01079
01080
                         .0040884, .0039088, .0037373, .0035739, .003418, .0032693,
01082
01083
                          .0031277, .0029926, .0028639, .0027413, .0026245, .0025133,
01084
                          .0024074, .0023066, .0022108, .0021196, .002033, .0019507,
                         .0018726, .0017985, .0017282, .0016617, .0015988, .0015394, .0014834, .0014306, .0013811, .0013346, .0012911, .0012506, .0012131, .0011784, .0011465, .0011175, .0010912, .0010678, .0012131, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010912, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .0010678, .00106
01085
01086
01087
                           .0010472, .0010295, .0010147, .001003, 9.9428e-4,
                                                                                                                                                          9.8883e-4
                          9.8673e-4, 9.8821e-4, 9.9343e-4, .0010027, .0010164, .0010348,
01089
01090
                         .0010586, .0010882, .0011245, .0011685, .0012145, .0012666,
01091
                          .0013095, .0013688, .0014048, .0014663, .0015309, .0015499,
                         .0016144, .0016312, .001705, .0017892, .0018499, .0019715, .0021102, .0022442, .0024284, .0025893, .0027703, .0029445,
01092
01093
                         .0031193, .003346, .0034552, .0036906, .0037584, .0040084, .0041934, .0044587, .0047093, .0049759, .0053421, .0055134,
01094
01095
01096
                          .0059048, .0058663, .0061036, .0063259, .0059657, .0060653,
                          .0060972, \ .0055539, \ .0055653, \ .0055772, \ .005331, \ .0054953,
01097
                         .0055919, .0058684, .006183, .0066675, .0069808, .0075142, .0078536, .0084282, .0089454, .0094625, .0093703, .0095857,
01098
01099
                         .0099283, .010063, .010521, .0097778, .0098175, .010379, .010447, .0105, .010617, .010706, .01078, .011177, .011212, .011304,
01101
01102
                          .011446, .011603, .011816, .012165, .012545, .013069, .013539,
                         .01411, .014776, .016103, .017016, .017994, .018978, .01998, .021799, .022745, .023681, .024627, .025562, .026992, .027958, .029013, .030154, .031402, .03228, .033651, .035272, .037088,
01103
01104
01105
```

```
.039021, .041213, .043597, .045977, .04877, .051809, .054943,
                .058064, .061528, .06537, .069309, .071928, .075752, .079589, .083352, .084096, .087497, .090817, .091198, .094966, .099045
01108
                .10429, .10867, .11518, .12269, .13126, .14087, .15161, .16388,
01109
01110
                .16423, .1759, .18721, .19994, .21275, .22513, .23041, .24231,
                .25299, .25396, .26396, .27696, .27929, .2908, .30595, .31433,
01111
                . 3282, . 3429, . 35944, . 37467, . 39277, . 41245, . 43326, . 45649, . 48152, . 51897, . 54686, . 57877, . 61263, . 64962, . 68983, . 73945,
01112
01113
                 .78619, .83537, .89622, .95002, 1.0067, 1.0742, 1.1355, 1.2007,
01114
                1.2738, 1.347, 1.4254, 1.5094, 1.6009, 1.6976, 1.8019, 1.9148, 2.0357, 2.166, 2.3066, 2.4579, 2.6208, 2.7966, 2.986, 3.188,
01115
01116
                3.4081, 3.6456, 3.9, 4.1747, 4.4712, 4.7931, 5.1359, 5.5097, 5.9117, 6.3435, 6.8003, 7.3001, 7.8385, 8.3945, 9.011, 9.6869, 10.392, 11.18, 12.036, 12.938, 13.944, 14.881, 16.029, 17.255,
01117
01119
01120
                18.574, 19.945, 21.38, 22.9, 24.477, 26.128, 27.87, 29.037,
                30.988, 33.145, 35.506, 37.76, 40.885, 44.487, 48.505, 52.911, 57.56, 61.964, 67.217, 72.26, 78.343, 85.08, 91.867, 99.435, 107.68, 116.97, 127.12, 138.32, 150.26, 163.04, 174.81, 189.26, 205.61, 224.68, 240.98, 261.88, 285.1, 307.58, 334.35, 363.53,
01121
01122
01123
                394.68, 427.85, 458.85, 489.25, 472.87, 486.93, 496.27, 501.52,
                501.57, 497.14, 488.09, 476.32, 393.76, 388.51, 393.42, 414.45,
01126
01127
                455.12, 514.62, 520.38, 547.42, 562.6, 487.47, 480.83, 391.06,
                376.92, 303.7, 295.91, 256.03, 236.73, 280.38, 310.71, 335.53,
01128
                367.88, 401.94, 435.52, 469.13, 497.94, 588.82, 597.94, 597.2, 588.28, 571.2, 555.75, 603.56, 638.15, 680.75, 801.72, 848.01, 962.15, 990.06, 1068.1, 1076.2, 1115.3, 1134.2, 1136.6, 1119.1,
01129
01130
01131
                1108.9, 1090.6, 1068.7, 1041.9, 1005.4, 967.98, 927.08, 780.1,
01132
01133
                751.41, 733.12, 742.65, 785.56, 855.16, 852.45, 878.1, 784.59,
                777.81, 765.13, 622.93, 498.09, 474.89, 386.9, 378.48, 336.17, 322.04, 329.57, 350.5, 383.38, 420.02, 462.39, 499.71, 531.98, 654.99, 653.43, 639.99, 605.16, 554.16, 504.42, 540.64, 552.33, 679.46, 699.51, 713.91, 832.17, 919.91, 884.96, 907.57, 846.56,
01134
01135
01136
01137
                818.56, 768.93, 706.71, 642.17, 575.95, 515.38, 459.07, 409.02, 364.61, 325.46, 291.1, 260.89, 234.39, 211.01, 190.38, 172.11,
01138
01139
                155.91, 141.49, 128.63, 117.13, 106.84, 97.584, 89.262, 81.756, 74.975, 68.842, 63.28, 58.232, 53.641, 49.46, 45.649, 42.168, 38.991, 36.078, 33.409, 30.96, 28.71, 26.642, 24.737, 22.985, 21.37, 19.882, 18.512, 17.242, 16.073, 14.987, 13.984, 13.05,
01140
01141
01142
01144
                12.186, 11.384, 10.637, 9.9436, 9.2988, 8.6991, 8.141, 7.6215,
                7.1378, 6.6872, 6.2671, 5.8754, 5.51, 5.1691, 4.851, 4.5539, 4.2764, 4.0169, 3.7742, 3.5472, 3.3348, 3.1359, 2.9495, 2.7749,
01145
01146
                2.6113, 2.4578, 2.3139, 2.1789, 2.0523, 1.9334, 1.8219, 1.7171, 1.6188, 1.5263, 1.4395, 1.3579, 1.2812, 1.209, 1.1411, 1.0773,
01147
01148
                1.0171, .96048, .90713, .85684, .80959, .76495, .72282, .68309,
01149
                .64563, .61035, .57707, .54573, .51622, .48834, .46199, .43709, .41359, .39129, .37034, .35064, .33198, .31442, .29784, .28218, .26732, .25337, .24017, .22774, .21601, .20479, .19426
01150
01151
01152
01153
01154
            static double co2260[2001] = { 5.7971e-5, 6.0733e-5, 6.3628e-5, 6.6662e-5,
01155
                6.9843e-5, 7.3176e-5, 7.6671e-5, 8.0334e-5, 8.4175e-5, 8.8201e-5,
                9.2421e-5, 9.6846e-5, 1.0149e-4, 1.0635e-4, 1.1145e-4, 1.1679e-4,
01157
01158
                1.224e-4, 1.2828e-4, 1.3444e-4, 1.409e-4, 1.4768e-4, 1.5479e-4,
                1.6224e-4, 1.7006e-4, 1.7826e-4, 1.8685e-4, 1.9587e-4, 2.0532e-4, 2.1524e-4, 2.2565e-4, 2.3656e-4, 2.48e-4, 2.6001e-4, 2.7261e-4,
01159
01160
                2.8582e-4, 2.9968e-4, 3.1422e-4, 3.2948e-4, 3.4548e-4, 3.6228e-4,
01161
                3.799e-4, 3.9838e-4, 4.1778e-4, 4.3814e-4, 4.595e-4, 4.8191e-4,
                5.0543e-4, 5.3012e-4, 5.5603e-4, 5.8321e-4, 6.1175e-4, 6.417e-4,
01163
                6.7314e-4, 7.0614e-4, 7.4078e-4, 7.7714e-4, 8.1531e-4, 8.5538e-4, 8.9745e-4, 9.4162e-4, 9.8798e-4, .0010367, .0010878, .0011415,
01164
01165
                .0011978, .001257, .0013191, .0013844, .001453, .0015249,
01166
                .0011978, .001257, .0013191, .0013844, .001453, .0015249, .0016006, .00168, .0017634, .001851, .001943, .0020397, .0021412, .0022479, .002247, .0024015, .0027316, .0028682, .0030117, .0031626, .0033211, .0034877, .0036628, .0038469, .0040403, .0042436, .0044574, .004682, .0049182, .0051665,
01167
01170
01171
                 .0054276, .0057021, .0059907, .0062942, .0066133, .0069489,
01172
                .0073018, .0076729, .0080632, .0084738, .0089056,
                                                                                                    .0093599.
                .0098377, .01034, .010869, .011426, .012011, .012627, .013276, .013958, .014676, .015431, .016226, .017063, .017944, .018872, .019848, .020876, .021958, .023098, .024298, .025561, .026892,
01173
01174
                .028293, .029769, .031323, .032961, .034686, .036503, .038418, .040435, .042561, .044801, .047161, .049649, .052271, .055035,
01176
01177
                .057948, .061019, .064256, .06767, .07127, .075066, .079069, .083291, .087744, .092441, .097396, .10262, .10814, .11396, .1201, .12658, .13342, .14064, .14826, .1563, .1648, .17376, .18323, .19324, .2038, .21496, .22674, .23919, .25234, .26624, .28093, .29646, .31287, .33021, .34855, .36794, .38844, .41012,
01178
01179
01180
01182
                01183
01184
01185
01186
                4.1866, 4.4437, 4.7174, 5.0089, 5.3192, 5.65, 6.0025, 6.3782, 6.7787, 7.206, 7.6617, 8.1479, 8.6669, 9.221, 9.8128, 10.445,
01189
01190
                11.12, 11.843, 12.615, 13.441, 14.325, 15.271, 16.283, 17.367
                18.529, 19.776, 21.111, 22.544, 24.082, 25.731, 27.504, 29.409, 31.452, 33.654, 36.024, 38.573, 41.323, 44.29, 47.492, 50.951,
01191
```

```
54.608, 58.588, 62.929, 67.629, 72.712, 78.226, 84.207, 90.699,
                     97.749, 105.42, 113.77, 122.86, 132.78, 143.61, 155.44, 168.33, 182.48, 198.01, 214.87, 233.39, 253.86, 276.34, 300.3, 327.28,
01194
01195
01196
                      356.89, 389.48, 422.29, 458.99, 501.39, 548.13, 595.62, 652.74,
                     716.54, 784.57, 866.78, 960.59, 1062.8, 1072.5, 1189.5, 1319.4, 1467.6, 1630.2, 1813.7, 2016.9, 2253., 2515.3, 2773.5, 3092.8, 3444.4, 3720.4, 4104.3, 4527.5, 4645.9, 5021.7, 5462.2, 5597.,
01197
01198
                      6110.6, 6732.5, 7513.8, 8270.6, 9640.6, 11487., 2796.1, 2680.1
01200
                     6110.6, 6732.5, 7513.8, 8270.6, 9640.6, 11487., 2796.1, 2680.1, 2441.6, 2404.2, 2334.8, 2215.2, 1642.5, 1477.9, 1328.1, 1223.5, 843.34, 766.96, 831.65, 834.84, 774.85, 1156.3, 1275.6, 1366.1, 1795.6, 1885., 1936.5, 1953.4, 2154.4, 2002.7, 1789.8, 10381., 9040., 8216.5, 7384.7, 6721.9, 6187.7, 6143.8, 5703.9, 5276.6, 4873.1, 4736., 4325.3, 3927., 3554.1, 3286.1, 2950.1, 2642.4, 2368.7, 2138.9, 1914., 1719.6, 1543.9, 1388.6, 1252.1, 1132.2,
01201
01202
01203
01204
01205
01206
01207
                      1024.1, 1025.4, 920.58, 829.59, 750.54, 685.01, 624.25, 570.14,
                     525.81, 481.85, 441.95, 408.71, 377.23, 345.86, 318.51, 292.26, 268.34, 247.04, 227.14, 209.02, 192.69, 177.59, 163.78, 151.26, 139.73, 129.19, 119.53, 110.7, 102.57, 95.109, 88.264, 81.948, 76.13, 70.768, 65.827, 61.251, 57.022, 53.495, 49.824, 46.443,
01208
01209
01210
                      43.307, 40.405, 37.716, 35.241, 32.923, 30.77, 28.78, 26.915,
01212
                     25.177, 23.56, 22.059, 20.654, 19.345, 18.126, 16.988, 15.93,
01213
01214
                     14.939, 14.014, 13.149, 12.343, 11.589, 10.884, 10.225, 9.6093,
                     9.0327, 8.4934, 7.9889, 7.5166, 7.0744, 6.6604, 6.2727, 5.9098,
01215
                     5.5701, 5.2529, 4.955, 4.676, 4.4148, 4.171, 3.9426, 3.7332, 3.5347, 3.3493, 3.1677, 3.0025, 2.8466, 2.6994, 2.5601, 2.4277,
01216
01218
                      2.3016, 2.1814, 2.0664, 1.9564, 1.8279, 1.7311, 1.6427, 1.5645,
                     1.4982, 1.443, 1.374, 1.3146, 1.2562, 1.17, 1.1105, 1.0272,
01219
01220
                      .96863, .89718, .83654, .80226, .75908, .72431, .69573, .67174,
                     .98863, .89/18, .83654, .80226, ./5908, ./2431, .69573, .6/174, .65126, .63315, .61693, .60182, .58715, .59554, .57649, .55526, .53177, .50622, .48176, .4813, .47642, .47492, .50273, .50293, .52687, .52239, .53419, .53814, .52626, .52211, .51492, .50622, .49746, .48841, .4792, .43534, .41999, .40349, .38586, .36799, .35108, .31089, .30803, .3171, .33599, .35041, .36149, .32924, .32462, .27309, .25961, .20922, .19504, .15683, .13098, .11588, .31049, .31262, .27309, .25961, .20922, .19504, .15683, .13098, .11588, .30803, .3174, .33549, .32462, .27309, .25961, .20922, .19504, .15683, .13098, .11588, .30803, .3174, .33642, .27385, .19084, .20226, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20227, .20
01221
01222
01223
01225
01226
01227
                      .11478, .11204, .11363, .12135, .16423, .17785, .19094, .20236,
                     .21084, .2154, .24108, .22848, .20871, .18797, .17963, .17834, .21552, .22284, .26945, .27052, .30108, .28977, .29772, .29224, .27658, .24956, .22777, .20654, .18392, .16338, .1452, .12916, .1152, .10304, .092437, .083163, .075031, .067878, .061564,
01228
01229
01231
                     01232
01233
                     .019136, .018141, .017278, .016541, .015926, .015432, .015058, .014807, .014666, .014635, .014728, .014947, .01527, .015728, .016345, .017026, .017798, .018839, .019752, .020636, .021886,
01234
01235
                     .022695, .02327, .023478, .024292, .023544, .022222, .021932, .020052, .018143, .017722, .017031, .017782, .01938, .020734,
01237
01238
                     .020476, .019255, .017477, .016878, .014617, .012489, .011765, .0099077, .0086446, .0079446, .0078644, .0079763, .008671, .01001, .0108, .012933, .015349, .016341, .018484, .020254, .020254, .020254, .020478, .019591, .018595, .018385, .019913, .022254, .024847, .025809, .028053, .029924, .030212, .031367, .03222,
01239
01240
01241
01242
                     .032739, .032537, .03286, .033344, .033507, .033499, .033339, .032809, .033041, .031723, .029837, .027511, .026603, .024032,
01244
01245
01246
                      .021914, .020948, .021701, .023425, .024259, .024987, .023818,
                     .021768, .019223, .018144, .015282, .012604, .01163, .0097907,
01247
                     .008336, .0082473, .0079582, .0088077, .009779, .010129, .01
.014378, .016761, .01726, .018997, .019998, .019809, .01819,
.016358, .016099, .01617, .017939, .020223, .022521, .02277,
01248
                                                                                                                                                   .012145.
01250
                     .024279, .025247, .024222, .023989, .023224, .021493, .020362, .018596, .017309, .015975, .014466, .013171, .011921, .01078, .007229, .0087612, .0078729, .0070682, .0063494, .0057156,
01251
01252
01253
                     .0051459, .0046273, .0041712, .0037686, .0034119, .003095, .0028126, .0025603, .0023342, .0021314, .0019489, .0017845,
01254
                      .001636, .0015017, .00138, .0012697, .0011694, .0010782,
01256
01257
                      9.9507e-4, 9.1931e-4, 8.5013e-4, 7.869e-4, 7.2907e-4, 6.7611e-4,
01258
                      6.2758e-4, 5.8308e-4, 5.4223e-4, 5.0473e-4, 4.7027e-4, 4.3859e-4,
01259
                     4.0946e-4, 3.8265e-4, 3.5798e-4, 3.3526e-4, 3.1436e-4, 2.9511e-4, 2.7739e-4, 2.6109e-4, 2.4609e-4, 2.3229e-4, 2.1961e-4, 2.0797e-4,
01260
                     1.9729e-4, 1.875e-4, 1.7855e-4, 1.7038e-4, 1.6294e-4, 1.5619e-4,
01261
                      1.5007e-4, 1.4456e-4, 1.3961e-4, 1.3521e-4, 1.3131e-4, 1.2789e-4,
                      1.2494e-4, 1.2242e-4, 1.2032e-4, 1.1863e-4, 1.1733e-4, 1.1641e-4,
01263
01264
                      1.1585e-4, 1.1565e-4, 1.158e-4, 1.1629e-4, 1.1712e-4, 1.1827e-4,
01265
                      1.1976e-4, 1.2158e-4, 1.2373e-4, 1.262e-4, 1.2901e-4, 1.3214e-4,
                      1.3562e-4, 1.3944e-4, 1.4361e-4, 1.4814e-4, 1.5303e-4, 1.5829e-4,
01266
                      1.6394e-4, 1.6999e-4, 1.7644e-4, 1.8332e-4, 1.9063e-4, 1.984e-4,
01267
                     2.0663e-4, 2.1536e-4, 2.246e-4, 2.3436e-4, 2.4468e-4, 2.5558e-4, 2.6708e-4, 2.7921e-4, 2.92e-4, 3.0548e-4, 3.1968e-4, 3.3464e-4,
01268
01269
                     3.5039e-4, 3.6698e-4, 3.8443e-4, 4.0281e-4, 4.2214e-4, 4.4248e-4, 4.6389e-4, 4.864e-4, 5.1009e-4, 5.3501e-4, 5.6123e-4, 5.888e-4, 6.1781e-4, 6.4833e-4, 6.8043e-4, 7.142e-4, 7.4973e-4, 7.8711e-4,
01270
01271
01272
                     8.2644e-4, 8.6783e-4, 9.1137e-4, 9.5721e-4, .0010054, .0010562,
01273
                      .0011096, .0011659, .0012251, .0012875, .0013532, .0014224,
                      .0014953, .001572, .0016529, .0017381, .0018279, .0019226,
01275
01276
                      .0020224, .0021277, .0022386, .0023557, .0024792, .0026095,
01277
                      .002747,\ .0028921,\ .0030453,\ .0032071,\ .003378,\ .0035586,
                     .0037494, .003951, .0041642, .0043897, .0046282, .0048805, .0051476, .0054304, .00573, .0060473, .0063837, .0067404,
01278
```

```
.0071188, .0075203, .0079466, .0083994, .0088806, .0093922,
               .0099366, .010516, .011134, .011792, .012494, .013244, .014046, .014898, .015808, .016781, .017822, .018929, .020108, .02138,
01281
01282
               .022729, .02419, .02576, .027412, .029233, .031198, .033301, .035594, .038092, .040767, .04372, .046918, .050246, .053974, .058009, .061976, .066586, .071537, .076209, .081856, .087998, .093821, .10113, .10913, .11731, .12724, .13821, .15025, .1639,
01283
01284
01285
                .17807, .19472, .21356, .23496, .25758, .28387,
                                                                                                .31389,
01288
                .37469, .40989, .43309, .46845, .5042, .5023, .52981, .55275,
01289
                .51075, .51976, .52457, .44779, .44721, .4503, .4243, .45244,
                .49491, .55399, .39021, .24802, .2501, .2618, .27475, .28879,
01290
                31317, 33643, 36257, 4018, 43275, 46525, 53333, 56599, 60557, 70142, 74194, 77736, 88567, 91182, 93294, 98407, 98772, 99176, 9995, 1.2405, 1.3602, 1.338, 1.3255, 1.3267, 1.3404, 1.3634, 1.3967, 1.4407, 1.4961, 1.5603, 1.6328, 1.7153,
01291
01292
01293
01294
                1.8094, 1.9091, 2.018, 2.1367, 2.264, 2.4035, 2.5562, 2.7179, 2.9017, 3.1052, 3.3304, 3.5731, 3.8488, 4.1553, 4.4769, 4.7818, 5.1711, 5.5204, 5.9516, 6.4097, 6.8899, 7.1118, 7.5469, 7.9735,
01295
01296
01297
                7.9511, 8.3014, 8.6418, 8.4757, 8.8256, 9.2294, 9.6923, 10.033, 10.842, 11.851, 11.78, 8.8435, 9.1381, 9.5956, 10.076, 10.629,
                11.22, 11.883, 12.69, 13.163, 13.974, 14.846, 16.027, 17.053,
01300
01301
                18.148, 19.715, 20.907, 22.163, 23.956, 25.235, 26.566, 27.94
01302
                29.576, 30.956, 32.432, 35.337, 39.911, 41.128, 42.625, 44.386,
                46.369, 48.619, 51.031, 53.674, 56.825, 59.921, 63.286, 66.929, 70.859, 75.081, 79.618, 84.513, 89.739, 95.335, 101.35, 107.76,
01303
01304
                114.63, 121.98, 129.87, 138.3, 147.34, 157.04, 167.56, 178.67,
                190.61, 203.43, 217.19, 231.99, 247.88, 264.98, 283.37, 303.17
01306
01307
                324.49, 347.47, 372.25, 398.98, 427.85, 459.06, 492.8, 529.31,
01308
                568.89, 611.79, 658.35, 708.91, 763.87, 823.65, 888.72, 959.58,
01309
                1036.8, 1121.8, 1213.9, 1314.3, 1423.8, 1543., 1672.8, 1813.4,
                1966.1, 2131.4, 2309.5, 2499.3, 2705., 2925.7, 3161.6, 3411.3,
01310
01311
                3611.5, 3889.2, 4191.1, 4519.3, 4877.9, 5272.9, 5712.9,
                6719.6, 7385., 8145., 8977.7, 9831.9, 10827., 11934., 13063.,
01312
01313
                14434., 15878., 17591., 19435., 21510., 23835., 26835., 29740.,
                32878., 36305., 39830., 43273., 46931., 50499., 49586., 51598., 53429., 54619., 55081., 55102., 54485., 53487., 52042., 42689.,
01314
01315
                42607., 44020., 47994., 54169., 53916., 55808., 56642., 46049., 44243., 32929., 30658., 21963., 20835., 15962., 13679., 17652., 19680., 22388., 25625., 29184., 32520., 35720., 38414., 40523.,
01316
01318
                49228., 48173., 45678., 41768., 37600., 41313., 42654., 44465.,
01319
01320
                55736., 56630., 65409., 63308., 66572., 61845., 60379., 56777.,
                51920., 46601., 41367., 36529., 32219., 28470., 25192., 22362.
01321
               19907., 17772., 15907., 14273., 12835., 11567., 10445., 9450.2, 8565.1, 7776., 7070.8, 6439.2, 5872.3, 5362.4, 4903., 4488.3, 4113.4, 3773.8, 3465.8, 3186.1, 2931.7, 2700.1, 2488.8, 2296., 2119.8, 1958.6, 1810.9, 1675.6, 1551.4, 1437.3, 1332.4, 1236.,
01322
01325
               1147.2, 1065.3, 989.86, 920.22, 855.91, 796.48, 741.53, 690.69, 643.62, 600.02, 559.6, 522.13, 487.35, 455.06, 425.08, 397.21, 371.3, 347.2, 324.78, 303.9, 284.46, 266.34, 249.45, 233.7, 219.01, 205.3, 192.5, 180.55, 169.38, 158.95, 149.2, 140.07,
01326
01327
01328
01329
                131.54, 123.56, 116.09, 109.09, 102.54, 96.405, 90.655, 85.266,
                80.213, 75.475, 71.031, 66.861, 62.948, 59.275, 55.827, 52.587,
01331
01332
                49.544, 46.686, 43.998, 41.473, 39.099, 36.867, 34.768, 32.795
01333
                30.939, 29.192, 27.546, 25.998, 24.539, 23.164, 21.869, 20.65, 19.501, 18.419, 17.399, 16.438, 15.532, 14.678, 13.874, 13.115,
01334
                12.4, 11.726, 11.088, 10.488, 9.921, 9.3846, 8.8784, 8.3996, 7.9469, 7.5197, 7.1174, 6.738, 6.379, 6.0409, 5.7213, 5.419, 5.1327, 4.8611, 4.6046, 4.3617, 4.1316, 3.9138, 3.7077, 3.5125,
01335
01337
                3.3281, 3.1536, 2.9885, 2.8323, 2.6846, 2.5447, 2.4124, 2.2871,
01338
01339
                2.1686, 2.0564, 1.9501, 1.8495, 1.7543, 1.6641, 1.5787, 1.4978,
                1.4212, 1.3486, 1.2799, 1.2147, 1.1529, 1.0943, 1.0388, .98602,
01340
               .93596, .8886, .84352, .80078, .76029, .722, .68585, .65161, .61901, .58808, .55854, .53044, .5039, .47853, .45459, .43173,
01341
                .41008, .38965, .37021, .35186, .33444, .31797, .30234, .28758, .2736, .26036, .24764, .2357, .22431, .21342, .20295, .19288,
01343
01344
01345
                .18334, .17444, .166, .15815, .15072, .14348, .13674, .13015,
               .1239, .11807, .11231, .10689, .10164, .096696, .091955, .087476, .083183, .079113, .075229, .071536, .068026, .064698, .06154, .058544, .055699, .052997, .050431, .047993, .045676, .043475, .041382, .039392, .037501, .035702, .033991, .032364,
01346
01347
01348
                .030817, .029345, .027945, .026613, .025345, .024139, .022991,
01350
01351
                .021899, .02086, .019871, .018929, .018033, .01718, .016368,
                .015595, .014859, .014158, .013491, .012856, .012251, .011675, .01126, .010604, .010107, .0096331, .009182, .0087523, .0083431,
01352
01353
                .0079533, .0075821, .0072284, .0068915, .0065706, .0062649, .0059737, .0056963, .005432, .0051802, .0049404, .0047118, .0044941, .0042867, .0040891, .0039009, .0037216, .0035507,
01354
01356
                .003388, .0032329, .0030852, .0029445, .0028105, .0026829, .0025613, .0024455, .0023353, .0022303, .0021304, .0020353,
01357
01358
                .0019448, .0018587, .0017767, .0016988, .0016247, .0015543,
01359
                .0014874, .0014238, .0013635, .0013062, .0012519, .0012005, .0011517, .0011057, .0010621, .001021, 9.8233e-4, 9.4589e-4, 9.1167e-4, 8.7961e-4, 8.4964e-4, 8.2173e-4, 7.9582e-4, 7.7189e-4, 7.499e-4, 7.2983e-4, 7.1167e-4, 6.9542e-4, 6.8108e-4, 6.6866e-4,
01360
01362
01363
01364
                6.5819e-4, 6.4971e-4, 6.4328e-4, 6.3895e-4, 6.3681e-4, 6.3697e-4,
                6.3956e-4, 6.4472e-4, 6.5266e-4, 6.6359e-4, 6.778e-4, 6.9563e-4, 7.1749e-4, 7.4392e-4, 7.7556e-4, 8.1028e-4, 8.4994e-4, 8.8709e-4,
01365
01366
```

```
9.3413e-4, 9.6953e-4, .0010202, .0010738, .0010976, .0011507,
                 .0011686, .0012264, .001291, .0013346, .0014246, .0015293, .0016359, .0017824, .0019255, .0020854, .002247, .0024148,
01368
01369
01370
                  .0026199, .0027523, .0029704, .0030702, .0033047, .0035013,
                 .0037576, .0040275, .0043089, .0046927, .0049307, .0053486, .0053809, .0056699, .0059325, .0055488, .005634, .0056392, .004946, .0048855, .0048208, .0044386, .0045498, .0046377,
01371
01372
                  .0048939, \ .0052396, \ .0057324, \ .0060859, \ .0066906, \ .0071148
01374
                 .0048939, .0052396, .0057324, .0060859, .006906, .0071148, .0077224, .0082687, .008769, .008471, .008572, .0087729, .008775, .0090742, .0080704, .0080288, .0085747, .0086087, .0086408, .0088752, .0089381, .0089757, .0093532, .0092824, .0092566, .0092645, .0092735, .009342, .0095806, .0097991, .010213, .010611, .011129, .011756, .013237, .01412, .015034, .015936, .01682, .018597, .019315, .019995, .020658, .021289, .022363, .022996, .023716, .024512, .025434, .026067, .027118, .028386, .02885, .031442, .033253, .03525, .037296, .03701
01375
01376
01377
01378
01380
01381
                  .028396, .029865, .031442, .033253, .03525, .037296, .039701, .042356, .045154, .048059, .051294, .054893, .058636, .061407
01382
01383
                 .065172, .068974, .072676, .073379, .076547, .079556, .079134, .082308, .085739, .090192, .09359, .099599, .10669, .11496,
01384
                  .1244, .13512, .14752, .14494, .15647, .1668, .17863, .19029
01386
                  .20124, .20254, .21179, .21982, .21625, .22364, .23405, .23382,
01387
01388
                  .2434, .25708, .26406, .27621, .28909, .30395, .31717, .33271,
                  .3496, .36765, .38774, .40949, .446, .46985, .49846, .5287, .562, .59841, .64598, .68834, .7327, .78978, .8373, .88708, .94744, 1.0006, 1.0574, 1.1215, 1.1856, 1.2546, 1.3292, 1.4107, 1.4974,
01389
01390
01391
                  1.5913, 1.6931, 1.8028, 1.9212, 2.0492, 2.1874, 2.3365, 2.4978, 2.6718, 2.8588, 3.062, 3.2818, 3.5188, 3.7752, 4.0527, 4.3542,
01392
01393
01394
                  4.6782, 5.0312, 5.4123, 5.8246, 6.2639, 6.7435, 7.2636, 7.8064,
                 8.4091, 9.0696, 9.7677, 10.548, 11.4, 12.309, 13.324, 14.284, 15.445, 16.687, 18.019, 19.403, 20.847, 22.366, 23.925, 25.537, 27.213, 28.069, 29.864, 31.829, 33.988, 35.856, 38.829, 42.321, 46.319, 50.606, 55.126, 59.126, 64.162, 68.708, 74.615, 81.176,
01395
01396
01397
01398
                  87.739, 95.494, 103.83, 113.38, 123.99, 135.8, 148.7, 162.58,
01399
01400
                  176.32, 192.6, 211.47, 232.7, 252.64, 277.41, 305.38, 333.44,
                  366.42, 402.66, 442.14, 484.53, 526.42, 568.15, 558.78, 582.6, 600.98, 613.94, 619.44, 618.24, 609.84, 595.96, 484.86, 475.59,
01401
01402
                  478.49, 501.56, 552.19, 628.44, 630.39, 658.92, 671.96, 562.7, 545.88, 423.43, 400.14, 306.59, 294.13, 246.8, 226.51, 278.21,
01403
                  314.39, 347.22, 389.13, 433.16, 477.48, 521.67, 560.54, 683.6, 696.37, 695.91, 683.1, 658.24, 634.89, 698.85, 742.87, 796.66, 954.49, 1009.5, 1150.5, 1179.1, 1267.9, 1272.4, 1312.7, 1330.4,
01405
01406
01407
                 1331.6, 1315.8, 1308.3, 1293.3, 1274.6, 1249.5, 1213.2, 1172.1, 1124.4, 930.33, 893.36, 871.27, 883.54, 940.76, 1036., 1025.6, 1053.1, 914.51, 894.15, 865.03, 670.63, 508.41, 475.15, 370.85,
01408
01409
01410
                  361.06, 319.38, 312.75, 331.87, 367.13, 415., 467.94, 525.49, 578.41, 624.66, 794.82, 796.97, 780.29, 736.49, 670.18, 603.75,
01412
                 578.41, 624.68, 794.62, 796.57, 786.29, 736.49, 670.16, 603.13, 659.67, 679.8, 857.12, 884.05, 900.65, 1046.1, 1141.9, 1083., 1089.2, 1e3, 947.08, 872.31, 787.91, 704.75, 624.93, 553.68, 489.91, 434.21, 385.64, 343.3, 306.42, 274.18, 245.94, 221.11, 199.23, 179.88, 162.73, 147.48, 133.88, 121.73, 110.86, 101.1, 92.323, 84.417, 77.281, 70.831, 64.991, 59.694, 54.884, 50.509,
01413
01414
01415
01416
                  46.526, 42.893, 39.58, 36.549, 33.776, 31.236, 28.907, 26.77,
01418
01419
                  24.805, 23., 21.339, 19.81, 18.404, 17.105, 15.909, 14.801,
                  13.778, 12.83, 11.954, 11.142, 10.389, 9.691, 9.0434, 8.4423, 7.8842, 7.3657, 6.8838, 6.4357, 6.0189, 5.6308, 5.2696, 4.9332, 4.6198, 4.3277, 4.0553, 3.8012, 3.5639, 3.3424, 3.1355, 2.9422,
01420
01421
01422
                  2.7614, 2.5924, 2.4343, 2.2864, 2.148, 2.0184, 1.8971, 1.7835, 1.677, 1.5773, 1.4838, 1.3961, 1.3139, 1.2369, 1.1645, 1.0966,
01424
01425
                  1.0329, .97309, .91686, .86406, .81439, .76767, .72381, .68252
01426
                  .64359, .60695, .57247, .54008, .50957, .48092, .45401, .42862,
                  .40465, .38202, .36072, .34052, .3216, .30386, .28711, .27135, .25651, .24252, .2293, .21689, .20517, .19416, .18381, .17396,
01427
01428
                  .16469
01430
01431
01432
              static double co2230[2001] = { 2.743e-5, 2.8815e-5, 3.027e-5, 3.1798e-5,
                 3.3405e-5, 3.5094e-5, 3.6869e-5, 3.8734e-5, 4.0694e-5, 4.2754e-5, 4.492e-5, 4.7196e-5, 4.9588e-5, 5.2103e-5, 5.4747e-5, 5.7525e-5,
01433
01434
01435
                  6.0446e-5, 6.3516e-5, 6.6744e-5, 7.0137e-5, 7.3704e-5, 7.7455e-5,
                  8.1397e-5, 8.5543e-5, 8.9901e-5, 9.4484e-5, 9.9302e-5, 1.0437e-4,
                  1.097e-4, 1.153e-4, 1.2119e-4, 1.2738e-4, 1.3389e-4, 1.4074e-4,
01437
01438
                  1.4795e-4, 1.5552e-4, 1.6349e-4, 1.7187e-4, 1.8068e-4, 1.8995e-4,
                  1.997e-4, 2.0996e-4, 2.2075e-4, 2.321e-4, 2.4403e-4, 2.5659e-4, 2.698e-4, 2.837e-4, 2.9832e-4, 3.137e-4, 3.2988e-4, 3.4691e-4,
01439
01440
                  3.6483e-4, 3.8368e-4, 4.0351e-4, 4.2439e-4, 4.4635e-4, 4.6947e-4, 4.9379e-4, 5.1939e-4, 5.4633e-4, 5.7468e-4, 6.0452e-4, 6.3593e-4,
01441
01442
                  6.69e-4, 7.038e-4, 7.4043e-4, 7.79e-4, 8.1959e-4, 8.6233e-4,
01443
01444
                  9.0732e-4, 9.5469e-4, .0010046, .0010571, .0011124, .0011706,
                  .0012319,\ .0012964,\ .0013644,\ .001436,\ .0015114,\ .0015908,
01445
                  .0016745, .0017625, .0018553, .0019531, .002056, .0021645, .0022788, .0023992, .002526, .0026596, .0028004, .0029488, .0031052, .0032699, .0034436, .0036265, .0038194, .0040227,
01446
01447
                  .0042369, .0044628, .0047008, .0049518, .0052164, .0054953,
01449
01450
                  .0057894, .0060995, .0064265, .0067713, .007135, .0075184,
                  .0079228, .0083494, .0087993, .0092738, .0097745, .010303, .01086, .011448, .012068, .012722, .013413, .014142, .014911, .015723, .01658, .017484, .018439, .019447, .020511, .021635,
01451
01452
01453
```

```
.022821, .024074, .025397, .026794, .02827, .029829, .031475,
                   .033215, .035052, .036994, .039045, .041213, .043504, .045926, .048485, .05119, .05405, .057074, .060271, .063651, .067225,
01455
01456
                    .071006, .075004, .079233, .083708, .088441, .093449, .098749,
01457
                   .10436, .11029, .11657, .12322, .13026, .13772, .14561, .15397, .16282, .1722, .18214, .19266, .20381, .21563, .22816, .24143, .2555, .27043, .28625, .30303, .32082, .3397, .35972, .38097,
01458
01459
                    .40352, .42746, .45286, .47983, .50847, .53888, .57119, .6055
01461
                   .40352, .42/46, .45286, .47983, .50847, .53888, .5/119, .6055, .64196, .6807, .72187, .76564, .81217, .86165, .91427, .97025, .0298, 1.0932, 1.1606, 1.2324, 1.3088, 1.3902, 1.477, 1.5693, 1.6678, 1.7727, 1.8845, 2.0038, 2.131, 2.2666, 2.4114, 2.5659, 2.7309, 2.907, 3.0951, 3.2961, 3.5109, 3.7405, 3.986, 4.2485, 4.5293, 4.8299, 5.1516, 5.4961, 5.8651, 6.2605, 6.6842, 7.1385, 7.6256, 8.1481, 8.7089, 9.3109, 9.9573, 10.652, 11.398, 12.2, 13.063, 13.992, 14.99, 16.064, 17.222, 18.469, 19.813, 21.263, 28.28, 24.516, 26.34, 28.31, 30.437, 32.738, 35.226, 37.914
01462
01463
01464
01465
01466
01467
01468
                   22.828, 24.516, 26.34, 28.31, 30.437, 32.738, 35.226, 37.914, 40.824, 43.974, 47.377, 51.061, 55.011, 59.299, 63.961, 69.013,
01469
01470
                   74.492, 80.444, 86.919, 93.836, 101.23, 109.25, 117.98, 127.47, 137.81, 149.07, 161.35, 174.75, 189.42, 205.49, 223.02, 242.26,
01471
                   263.45, 286.75, 311.94, 340.01, 370.86, 404.92, 440.44, 480.27, 525.17, 574.71, 626.22, 686.8, 754.38, 827.07, 913.38, 1011.7,
01474
                   121.5, 1161.6, 1289.5, 1432.2, 1595.4, 1777, 1983.3, 2216.1, 2485.7, 2788.3, 3101.5, 3481., 3902.1, 4257.1, 4740., 5272.8, 5457.9, 5946.2, 6505.3, 6668.4, 7302.4, 8061.6, 9015.8, 9908.3, 11613., 13956., 3249.6, 3243., 2901.5, 2841.3, 2729.6, 2558.2, 1797.8, 1583.2, 1386., 1233.5, 787.74, 701.46, 761.66, 767.21,
01475
01476
01477
01478
                    722.83, 1180.6, 1332.1, 1461.6, 2032.9, 2166., 2255.9, 2294.7,
01480
01481
                    2587.2, 2396.5, 2122.4, 12553., 10784., 9832.5, 8827.3, 8029.1,
01482
                    7377.9, 7347.1, 6783.8, 6239.1, 5721.1, 5503., 4975.1, 4477.8,
                   4021.3, 3676.8, 3275.3, 2914.9, 2597.4, 2328.2, 2075.4, 1857.6, 1663.6, 1493.3, 1343.8, 1213.3, 1095.6, 1066.5, 958.91, 865.15,
01483
01484
01485
                    783.31, 714.35, 650.77, 593.98, 546.2, 499.9, 457.87, 421.75,
                   387.61, 355.25, 326.62, 299.7, 275.21, 253.17, 232.83, 214.31, 197.5, 182.08, 167.98, 155.12, 143.32, 132.5, 122.58, 113.48,
01486
01487
                   105.11, 97.415, 90.182, 83.463, 77.281, 71.587, 66.341, 61.493, 57.014, 53.062, 49.21, 45.663, 42.38, 39.348, 36.547, 33.967, 31.573, 29.357, 27.314, 25.415, 23.658, 22.03, 20.524, 19.125, 17.829, 16.627, 15.511, 14.476, 13.514, 12.618, 11.786, 11.013,
01488
01489
01490
01492
                    10.294, 9.6246, 9.0018, 8.4218, 7.8816, 7.3783, 6.9092, 6.4719,
                   6.0641, 5.6838, 5.3289, 4.998, 4.6893, 4.4014, 4.1325, 3.8813, 3.6469, 3.4283, 3.2241, 3.035, 2.8576, 2.6922, 2.5348, 2.3896,
01493
01494
                   2.2535, 2.1258, 2.0059, 1.8929, 1.7862, 1.6854, 1.5898, 1.4992,
01495
                   1.4017, 1.3218, 1.2479, 1.1809, 1.1215, 1.0693, 1.0116, .96016, .9105, .84859, .80105, .74381, .69982, .65127, .60899, .57843, .54592, .51792, .49336, .47155, .45201, .43426, .41807, .40303, .38876, .3863, .37098, .35492, .33801, .32032, .30341, .29874, .29193, .28689, .29584, .29155, .29826, .29195, .29287, .2904,
01496
01497
01/99
01500
01501
                    .28199, .27709, .27162, .26622, .26133, .25676, .25235, .23137
                   .22365, .21519, .20597, .19636, .18699, .16485, .16262, .16643, .17542, .18198, .18631, .16759, .16338, .13505, .1267, .10053, .092554, .074093, .062159, .055523, .054849, .05401, .05528,
01502
01503
                    .058982, .07952, .08647, .093244, .099285, .10393, .10661,
01505
01506
                    .12072, .11417, .10396, .093265, .089137, .088909,
                                                                                                                         .10902
                   .11277, .13625, .13565, .14907, .14167, .1428, .13744, .12768, .11382, .10244, .091686, .08109, .071739, .063616, .056579,
01507
01508
                   .050504, .045251, .040689, .036715, .033237, .030181, .027488, .025107, .022998, .021125, .01946, .017979, .016661, .015489, .014448, .013526, .012712, .011998, .011375, .010839, .010384,
01509
01511
                    .010007, .0097053, .0094783, .0093257, .0092489, .0092504,
01512
                   .0093346, .0095077, .0097676, .01012, .01058, .011157, .011844, .012672, .013665, .014766, .015999, .017509, .018972, .020444, .022311, .023742, .0249, .025599, .026981, .026462, .025143, .025066, .022814, .020458, .020026, .019142, .020189, .022371,
01513
01514
01515
                   .024163, .023728, .02199, .019506, .018591, .015576, .012784,
01518
                    .011744, .0094777, .0079148, .0070652, .006986, .0071758,
01519
                    .008086, .0098025, .01087, .013609, .016764, .018137, .021061,
01520
                    .023498, .023576, .023965, .022828, .021519, .021283, .023364,
                   .026457, .029782, .030856, .033486, .035515, .035543, .036558, .037198, .037472, .037045, .037284, .03777, .038085, .038366, .038526, .038282, .038915, .037697, .035667, .032941, .031959,
01521
01522
                   .028692, .025918, .024596, .025592, .027873, .028935, .02984, .028148, .025305, .021912, .020454, .016732, .013357, .01205,
01524
01525
                   .009731, .0079881, .0077704, .0074387, .0083895, .0096776, .010326, .01293, .015955, .019247, .020145, .02267, .024231, .024184, .022131, .019784, .01955, .01971, .022119, .025116, .027978, .028107, .029808, .030701, .029164, .028551, .027286, .024946, .023259, .020982, .019221, .017471, .015643, .014074,
01526
01527
01528
01530
                    .01261, .011301, .010116, .0090582, .0081036, .0072542, .0065034, .0058436, .0052571, .0047321, .0042697, .0038607, .0034977,
01531
01532
                    .0031747, .0028864, .0026284, .002397, .002189, .0020017,
01533
                   .0018326, .0016798, .0015414, .0014159, .0013019, .0011983, .0011039, .0010177, 9.391e-4, 8.6717e-4, 8.0131e-4, 7.4093e-4,
01534
                   6.8553e-4, 6.3464e-4, 5.8787e-4, 5.4487e-4, 5.0533e-4, 4.69e-4, 4.3556e-4, 4.0474e-4, 3.7629e-4, 3.5e-4, 3.2569e-4, 3.032e-4,
01536
01537
01538
                   2.8239e-4, 2.6314e-4, 2.4535e-4, 2.2891e-4, 2.1374e-4, 1.9975e-4,
                   1.8685e-4, 1.7498e-4, 1.6406e-4, 1.5401e-4, 1.4479e-4, 1.3633e-4, 1.2858e-4, 1.2148e-4, 1.1499e-4, 1.0907e-4, 1.0369e-4, 9.8791e-5,
01539
```

```
9.4359e-5, 9.0359e-5, 8.6766e-5, 8.3555e-5, 8.0703e-5, 7.8192e-5,
              7.6003e-5, 7.4119e-5, 7.2528e-5, 7.1216e-5, 7.0171e-5, 6.9385e-5, 6.8848e-5, 6.8554e-5, 6.8496e-5, 6.8669e-5, 6.9069e-5, 6.9694e-5,
01542
01543
               7.054e-5, 7.1608e-5, 7.2896e-5, 7.4406e-5, 7.6139e-5, 7.8097e-5,
01544
              8.0283e-5, 8.2702e-5, 8.5357e-5, 8.8255e-5, 9.1402e-5, 9.4806e-5, 9.8473e-5, 1.0241e-4, 1.0664e-4, 1.1115e-4, 1.1598e-4, 1.2112e-4,
01545
01546
               1.2659e-4, 1.3241e-4, 1.3859e-4, 1.4515e-4, 1.521e-4, 1.5947e-4,
               1.6728e-4, 1.7555e-4, 1.8429e-4, 1.9355e-4, 2.0334e-4, 2.1369e-4,
01548
01549
               2.2463e-4, 2.3619e-4, 2.4841e-4, 2.6132e-4, 2.7497e-4, 2.8938e-4,
01550
              3.0462e-4, 3.2071e-4, 3.3771e-4, 3.5567e-4, 3.7465e-4, 3.947e-4,
               4.1588e-4, 4.3828e-4, 4.6194e-4, 4.8695e-4, 5.1338e-4, 5.4133e-4,
01551
               5.7087e-4, 6.0211e-4, 6.3515e-4, 6.701e-4, 7.0706e-4, 7.4617e-4,
01552
               7.8756e-4, 8.3136e-4, 8.7772e-4, 9.2681e-4, 9.788e-4,
                                                                                               .0010339.
              .0010922, .001154, .0012195, .0012889, .0013626, .0014407,
01554
01555
               .0015235, .0016114, .0017048, .0018038, .001909, .0020207,
01556
               .0021395, .0022657, .0023998, .0025426, .0026944, .002856,
               .0030281, .0032114, .0034068, .003615, .0038371, .004074,
01557
              .004327, .0045971, .0048857, .0051942, .0055239, .0058766, .0062538, .0066573, .0070891, .007551, .0080455, .0085747, .0091412, .0097481, .010397, .011092, .011837, .012638, .013495,
01558
01560
              .014415, .01541, .016475, .017621, .018857, .020175, .02162, .023185, .024876, .02672, .028732, .030916, .033319, .035939,
01561
01562
               .038736, .041847, .04524, .048715, .052678, .056977, .061203,
01563
              .066184, .07164, .076952, .083477, .090674, .098049, .10697,
01564
              . 1169, 1277, 14011, 15323, 1684, 18601, 20626, 22831, 25417, 28407, 31405, 34957, 38823, 41923, 46026, 50409, 51227, 54805, 57976, 53818, 55056, 557, 46741, 46403,
01565
01566
01567
01568
               .4636, .42265, .45166, .49852, .56663, .34306, .17779, .17697
              . 18346, . 19129, . 20014, . 21778, . 23604, . 25649, . 28676, . 31238, . 33856, . 39998, . 4288, . 46568, . 56654, . 60786, . 64473, . 76466, . 7897, . 80778, . 86443, . 85736, . 84798, . 84157, 1.1385, 1.2446, 1.1923, 1.1552, 1.1338, 1.1266, 1.1292, 1.1431, 1.1683, 1.2059,
01569
01570
01571
              1.2521, 1.3069, 1.3712, 1.4471, 1.5275, 1.6165, 1.7145, 1.8189, 1.9359, 2.065, 2.2007, 2.3591, 2.5362, 2.7346, 2.9515, 3.2021,
01573
01574
              3.4851, 3.7935, 4.0694, 4.4463, 4.807, 5.2443, 5.7178, 6.2231, 6.4796, 6.9461, 7.4099, 7.3652, 7.7182, 8.048, 7.7373, 8.0363, 8.3855, 8.8044, 9.0257, 9.8574, 10.948, 10.563, 6.8979, 7.0744,
01575
01576
01577
               7.4121, 7.7663, 8.1768, 8.6243, 9.1437, 9.7847, 10.182, 10.849,
01579
               11.572, 12.602, 13.482, 14.431, 15.907, 16.983, 18.11, 19.884,
               21.02, 22.18, 23.355, 24.848, 25.954, 27.13, 30.186, 34.893,
01580
01581
              35.682, 36.755, 38.111, 39.703, 41.58, 43.606, 45.868, 48.573
              51.298, 54.291, 57.559, 61.116, 64.964, 69.124, 73.628, 78.471, 83.683, 89.307, 95.341, 101.84, 108.83, 116.36, 124.46, 133.18, 142.57, 152.79, 163.69, 175.43, 188.11, 201.79, 216.55, 232.51,
01582
01583
              249.74, 268.38, 288.54, 310.35, 333.97, 359.55, 387.26, 417.3, 449.88, 485.2, 523.54, 565.14, 610.28, 659.31, 712.56, 770.43,
01586
01587
              833.36, 901.82, 976.36, 1057.6, 1146.8, 1243.8, 1350., 1466.3,
01588
              1593.6, 1732.7, 1884.1, 2049.1, 2228.2, 2421.9, 2629.4, 2853.7,
               3094.4, 3351.1, 3622.3, 3829.8, 4123.1, 4438.3, 4777.2, 5144.1,
01589
              5545.4, 5990.5, 6404.5, 6996.8, 7687.6, 8482.9, 9349.4, 10203.,
01590
               11223., 12358., 13493., 14916., 16416., 18236., 20222., 22501.,
               25102., 28358., 31707., 35404., 39538., 43911., 48391., 53193.,
01592
01593
               58028., 58082., 61276., 64193., 66294., 67480., 67921., 67423.,
01594
               66254., 64341., 51737., 51420., 53072., 58145., 66195., 65358.,
               67377., 67869., 53509., 50553., 35737., 32425., 21704., 19974.,
01595
              14457., 12142., 16798., 19489., 23049., 27270., 31910., 36457., 40877., 44748., 47876., 59793., 58626., 55454., 50337., 44893.,
01596
               50228., 52216., 54747., 69541., 70455., 81014., 77694., 80533.,
01598
01599
               73953., 70927., 65539., 59002., 52281., 45953., 40292., 35360.,
              31124., 27478., 24346., 21647., 19308., 17271., 15491., 13927., 12550., 11331., 10250., 9288.8, 8431.4, 7664.9, 6978.3, 6361.8, 5807.4, 5307.7, 4856.8, 4449., 4079.8, 3744.9, 3440.8, 3164.2, 2912.3, 2682.7, 2473., 2281.4, 2106., 1945.3, 1797.9, 1662.5,
01600
01601
01602
              1538.1, 1423.6, 1318.1, 1221., 1131.5, 1049., 972.99, 902.87, 838.01, 777.95, 722.2, 670.44, 622.35, 577.68, 536.21, 497.76,
01604
01605
01606
               462.12, 429.13, 398.61, 370.39, 344.29, 320.16, 297.85, 277.2,
              258.08, 240.38, 223.97, 208.77, 194.66, 181.58, 169.43, 158.15, 147.67, 137.92, 128.86, 120.44, 112.6, 105.3, 98.499, 92.166,
01607
01608
              86.264, 80.763, 75.632, 70.844, 66.381, 62.213, 58.321, 54.685, 51.288, 48.114, 45.145, 42.368, 39.772, 37.341, 35.065, 32.937,
01609
               30.943, 29.077, 27.33, 25.693, 24.158, 22.717, 21.367, 20.099,
01611
01612
              18.909, 17.792, 16.744, 15.761, 14.838, 13.971, 13.157, 12.393,
              11.676, 11.003, 10.369, 9.775, 9.2165, 8.6902, 8.1963, 7.7314, 7.2923, 6.8794, 6.4898, 6.122, 5.7764, 5.4525, 5.1484, 4.8611,
01613
01614
              4.5918, 4.3379, 4.0982, 3.8716, 3.6567, 3.4545, 3.2634, 3.0828, 2.9122, 2.7512, 2.5993, 2.4561, 2.3211, 2.1938, 2.0737, 1.9603,
01615
01616
              1.8534, 1.7525, 1.6572, 1.5673, 1.4824, 1.4022, 1.3265, 1.2551,
01617
              1.1876, 1.1239, 1.0637, 1.0069, .9532, .90248, .85454, .80921, .76631, .72569, .6872, .65072, .61635, .5836, .55261, .52336, .49581, .46998, .44559, .42236, .40036, .37929, .35924, .34043,
01618
01619
01620
              32238, 30547, 28931, 27405, 25975, 24616, 23341, 22133, 20997, 19924, 18917, 17967, 17075, 16211, 15411, 14646, 13912, 13201, 12509, 11857, 11261, 10698, 10186, 097039,
01621
01623
01624
               .092236, .087844, .083443, .07938, .075452, .071564, .067931,
01625
               .064389, .061078, .057901, .054921, .052061, .049364, .046789,
              .04435, .042044, .039866, .037808, .035863, .034023, .032282, .030634, .029073, .027595, .026194, .024866, .023608, .022415,
01626
01627
```

```
.021283, .02021, .019193, .018228, .017312, .016443, .015619,
              .014837, .014094, .01339, .012721, .012086, .011483, .010
.010368, .009852, .0093623, .0088972, .0084556, .0080362,
01629
01630
01631
               .0076379, .0072596, .0069003, .006559, .0062349, .0059269,
              .0056344, .0053565, .0050925, .0048417, .0046034, .004377, .0041618, .0039575, .0037633, .0035788, .0034034, .0032368,
01632
01633
              .0030785, .002928, .0027851, .0026492, .0025201, .0023975, .0022809, .0021701, .0020649, .0019649, .0018699, .0017796,
01635
01636
               .0016938, .0016122, .0015348, .0014612, .0013913, .001325,
              .0012619, .0012021, .0011452, .0010913, .0010401, 9.9149e-4, 9.454e-4, 9.0169e-4, 8.6024e-4, 8.2097e-4, 7.8377e-4, 7.4854e-4,
01637
01638
              7.1522e-4, 6.8371e-4, 6.5393e-4, 6.2582e-4, 5.9932e-4, 5.7435e-4, 5.5087e-4, 5.2882e-4, 5.0814e-4, 4.8881e-4, 4.7076e-4, 4.5398e-4,
01639
01640
               4.3843e-4, 4.2407e-4, 4.109e-4, 3.9888e-4, 3.88e-4, 3.7826e-4
01641
01642
              3.6963e-4, 3.6213e-4, 3.5575e-4, 3.505e-4, 3.464e-4, 3.4346e-4,
01643
               3.4173e-4, 3.4125e-4, 3.4206e-4, 3.4424e-4, 3.4787e-4, 3.5303e-4,
01644
              3.5986e-4, 3.6847e-4, 3.7903e-4, 3.9174e-4, 4.0681e-4, 4.2455e-4,
              4.4527e-4, 4.6942e-4, 4.9637e-4, 5.2698e-4, 5.5808e-4, 5.9514e-4, 6.2757e-4, 6.689e-4, 7.1298e-4, 7.3955e-4, 7.8403e-4, 8.0449e-4, 8.5131e-4, 9.0256e-4, 9.3692e-4, .0010051, .0010846, .0011678,
01645
01646
              .001282, .0014016, .0015355, .0016764, .0018272, .0020055,
01648
01649
               .0021455, .0023421, .0024615, .0026786, .0028787, .0031259
01650
               .0034046, .0036985, .0040917, .0043902, .0048349, .0049531,
              .0052989, .0056148, .0052452, .0053357, .005333, .0045069, .0043851, .004253, .003738, .0038084, .0039013, .0041505, .0045372, .0050569, .0054507, .0061267, .0066122, .0072449, .0078012, .0082651, .0076538, .0076573, .0076806, .0075227,
01651
01652
01653
01654
01655
               .0076269, .0063758, .006254, .0067749, .0067909, .0068231,
01656
               .0072143, .0072762, .0072954, .007679, .0075107, .0073658,
              .0072441, .0071074, .0070378, .007176, .0072472, .0075844, .0079291, .008412, .0090165, .010688, .011535, .012375, .013166, .013895, .015567, .016011, .016392, .016737, .017043, .017731, .018031, .018419, .018877, .019474, .019868, .020604, .021538,
01657
01658
01659
01660
01661
              .022653, .023869, .025288, .026879, .028547, .030524, .03274,
              .035132, .03769, .040567, .043793, .047188, .049962, .053542, .057205, .060776, .061489, .064419, .067124, .065945, .068487
01662
01663
              .071209, .074783, .077039, .082444, .08902, .09692, .10617, .11687, .12952, .12362, .13498, .14412, .15492, .16519, .1744, .17096, .17714, .18208, .17363, .17813, .18564, .18295, .19045,
01664
01665
               .20252, .20815, .21844, .22929, .24229, .25321, .26588, .2797,
01667
01668
               .29465, .31136, .32961, .36529, .38486, .41027, .43694, .4667,
              .49943, .54542, .58348, .62303, .67633, .71755, .76054, .81371, .85934, .90841, .96438, 1.0207, 1.0821, 1.1491, 1.2226, 1.3018, 1.388, 1.4818, 1.5835, 1.6939, 1.8137, 1.9435, 2.0843, 2.237,
01669
01670
01671
              2.4026, 2.5818, 2.7767, 2.9885, 3.2182, 3.4679, 3.7391, 4.0349, 4.3554, 4.7053, 5.0849, 5.4986, 5.9436, 6.4294, 6.9598, 7.5203,
01673
01674
               8.143, 8.8253, 9.5568, 10.371, 11.267, 12.233, 13.31, 14.357,
              15.598, 16.93, 18.358, 19.849, 21.408, 23.04, 24.706, 26.409, 28.153, 28.795, 30.549, 32.43, 34.49, 36.027, 38.955, 42.465, 46.565, 50.875, 55.378, 59.002, 63.882, 67.949, 73.693, 80.095, 86.403, 94.264, 102.65, 112.37, 123.3, 135.54, 149.14, 163.83,
01675
01676
01677
               179.17, 196.89, 217.91, 240.94, 264.13, 292.39, 324.83, 358.21,
01679
01680
               397.16, 440.5, 488.6, 541.04, 595.3, 650.43, 652.03, 688.74,
              719.47, 743.54, 757.68, 762.35, 756.43, 741.42, 595.43, 580.97, 580.83, 605.68, 667.88, 764.49, 759.93, 789.12, 798.17, 645.66,
01681
01682
              615.65, 455.05, 421.09, 306.45, 289.14, 235.7, 215.52, 274.57, 316.53, 357.73, 409.89, 465.06, 521.84, 579.02, 630.64, 794.46
01683
               813., 813.56, 796.25, 761.57, 727.97, 812.14, 866.75, 932.5,
01685
               1132.8, 1194.8, 1362.2, 1387.2, 1482.3, 1479.7, 1517.9, 1533.1,
01686
01687
               1534.2, 1523.3, 1522.5, 1515.5, 1505.2, 1486.5, 1454., 1412.,
              1358.8, 1107.8, 1060.9, 1033.5, 1048.2, 1122.4, 1248.9, 1227.1, 1255.4, 1058.9, 1020.7, 970.59, 715.24, 512.56, 468.47, 349.3, 338.26, 299.22, 301.26, 332.38, 382.08, 445.49, 515.87, 590.85,
01688
01689
               662.3, 726.05, 955.59, 964.11, 945.17, 891.48, 807.11, 720.9,
01692
               803.36, 834.46, 1073.9, 1107.1, 1123.6, 1296., 1393.7, 1303.1,
01693
              1284.3, 1161.8, 1078.8, 976.13, 868.72, 767.4, 674.72, 593.73,
              523.12, 462.24, 409.75, 364.34, 325., 290.73, 260.76, 234.46, 211.28, 190.78, 172.61, 156.44, 142.01, 129.12, 117.57, 107.2, 97.877, 89.47, 81.882, 75.021, 68.807, 63.171, 58.052, 53.396,
01694
01695
01696
               49.155, 45.288, 41.759, 38.531, 35.576, 32.868, 30.384, 28.102,
               26.003, 24.071, 22.293, 20.655, 19.147, 17.756, 16.476, 15.292,
01698
01699
               14.198, 13.183, 12.241, 11.367, 10.554, 9.7989, 9.0978, 8.4475
              7.845, 7.2868, 6.7704, 6.2927, 5.8508, 5.4421, 5.064, 4.714, 4.3902, 4.0902, 3.8121, 3.5543, 3.315, 3.093, 2.8869, 2.6953,
01700
01701
               2.5172, 2.3517, 2.1977, 2.0544, 1.9211, 1.7969, 1.6812, 1.5735,
01702
               1.4731, 1.3794, 1.2921, 1.2107, 1.1346, 1.0637, .99744, .93554,
01703
               .87771, .82368, .77313, .72587, .6816, .64014, .60134, .565,
01704
01705
               .53086, .49883, .46881, .44074, .4144, .38979, .36679, .34513
01706
               .32474, .30552, .28751, .27045, .25458, .23976, .22584, .21278,
01707
               .20051, .18899, .17815, .16801, .15846, .14954, .14117, .13328,
01708
               .12584
01710
01711
           double xw, dw, ew, cw296, cw260, cw230, dt230, dt260, dt296, ctw, ctmpth;
01712
01713
           int iw, jw;
01714
```

```
/* Get CO2 continuum absorption... */
          xw = nu / 2 + 1;
if (xw >= 1 && xw < 2001) {
01716
01717
            iw = (int) xw;
01718
             jw = iw + 1;

dw = xw - iw;
01719
01720
             ew = 1 - dw;
01721
             cw296 = ew * co2296[iw - 1] + dw * co2296[jw - 1];
cw260 = ew * co2260[iw - 1] + dw * co2260[jw - 1];
cw230 = ew * co2230[iw - 1] + dw * co2230[jw - 1];
01722
01723
01724
             dt230 = t - 230;
01725
             dt260 = t - 260;
01726
             dt296 = t - 296;
01727
             ctw = dt260 * 5.050505e-4 * dt296 * cw230 - dt230 * 9.259259e-4
  * dt296 * cw260 + dt230 * 4.208754e-4 * dt260 * cw296;
01728
01729
01730
              ctmpth = u / GSL_CONST_NUM_AVOGADRO / 1000 * p / P0 * ctw;
01731
          } else
01732
            ctmpth = 0;
           return ctmpth;
01734 }
01735
01737
01738 double ctmh2o(
01739
          double nu,
01740
          double p,
01741
           double t,
          double q,
01742
01743
          double u)
01744
          static double h2o296[2001] = { .17, .1695, .172, .168, .1687, .1624, .1606, .1508, .1447, .1344, .1214, .1133, .1009, .09217, .08297, .06989,
01745
             .06513, .05469, .05056, .04417, .03779, .03484, .02994, .0272, .02325, .02063, .01818, .01592, .01405, .01251, .0108, .009647,
01747
01748
             .008424, .007519, .006555, .00588, .005136, .004511, .003889, .003509, .003114, .00274, .002446, .002144, .001895, .001676, .001486, .001312, .001164, .001031, 9.129e-4, 8.106e-4, 7.213e-4, 6.4e-4, 5.687e-4, 5.063e-4, 4.511e-4, 4.029e-4, 3.596e-4,
01749
01750
01751
01752
01753
              3.22e-4, 2.889e-4, 2.597e-4, 2.337e-4, 2.108e-4, 1.907e-4,
01754
              1.728e-4, 1.57e-4, 1.43e-4, 1.305e-4, 1.195e-4, 1.097e-4,
01755
             1.009e-4, 9.307e-5, 8.604e-5, 7.971e-5, 7.407e-5, 6.896e-5,
              6.433e-5, 6.013e-5, 5.631e-5, 5.283e-5, 4.963e-5, 4.669e-5,
01756
             4.398e-5, 4.148e-5, 3.917e-5, 3.702e-5, 3.502e-5, 3.316e-5, 3.142e-5, 2.978e-5, 2.825e-5, 2.681e-5, 2.546e-5, 2.419e-5,
01757
01758
              2.299e-5, 2.186e-5, 2.079e-5, 1.979e-5, 1.884e-5, 1.795e-5,
01759
01760
             1.711e-5, 1.633e-5, 1.559e-5, 1.49e-5, 1.426e-5, 1.367e-5,
01761
             1.312e-5, 1.263e-5, 1.218e-5, 1.178e-5, 1.143e-5, 1.112e-5,
             1.088e-5, 1.07e-5, 1.057e-5, 1.05e-5, 1.051e-5, 1.059e-5, 1.076e-5, 1.1e-5, 1.133e-5, 1.18e-5, 1.237e-5, 1.308e-5,
01762
01763
             1.393e-5, 1.483e-5, 1.614e-5, 1.758e-5, 1.93e-5, 2.123e-5, 2.346e-5, 2.647e-5, 2.93e-5, 3.279e-5, 3.745e-5, 4.152e-5,
01764
01766
              4.813e-5, 5.477e-5, 6.203e-5, 7.331e-5, 8.056e-5, 9.882e-5,
01767
             1.05e-4, 1.21e-4, 1.341e-4, 1.572e-4, 1.698e-4, 1.968e-4,
             1.036-4, 1.216-4, 1.346-4, 1.036-4, 1.036-4, 1.036-4, 1.036-4, 2.175e-4, 2.431e-4, 2.735e-4, 2.867e-4, 3.19e-4, 3.371e-4, 3.554e-4, 3.726e-4, 3.837e-4, 3.878e-4, 3.864e-4, 3.858e-4, 3.841e-4, 3.852e-4, 3.815e-4, 3.762e-4, 3.618e-4, 3.579e-4, 3.45e-4, 3.202e-4, 3.018e-4, 2.785e-4, 2.602e-4, 2.416e-4,
01768
01769
01770
01771
01772
             2.097e-4, 1.939e-4, 1.689e-4, 1.498e-4, 1.308e-4, 1.17e-4,
01773
             1.011e-4, 9.237e-5, 7.909e-5, 7.006e-5, 6.112e-5, 5.401e-5
01774
              4.914e-5, 4.266e-5, 3.963e-5, 3.316e-5, 3.037e-5, 2.598e-5,
             2.294e-5, 2.066e-5, 1.813e-5, 1.583e-5, 1.423e-5, 1.247e-5, 1.116e-5, 9.76e-6, 8.596e-6, 7.72e-6, 6.825e-6, 6.108e-6,
01775
01776
              5.366e-6, 4.733e-6, 4.229e-6, 3.731e-6, 3.346e-6, 2.972e-6,
01778
              2.628e-6, 2.356e-6, 2.102e-6, 1.878e-6, 1.678e-6, 1.507e-6,
             1.348e-6, 1.21e-6, 1.089e-6, 9.806e-7, 8.857e-7, 8.004e-7, 7.261e-7, 6.599e-7, 6.005e-7, 5.479e-7, 5.011e-7, 4.595e-7,
01779
01780
             4.219e-7, 3.885e-7, 3.583e-7, 3.314e-7, 3.071e-7, 2.852e-7, 2.654e-7, 2.474e-7, 2.311e-7, 2.162e-7, 2.026e-7, 1.902e-7,
01781
01782
             1.788e-7, 1.683e-7, 1.587e-7, 1.497e-7, 1.415e-7, 1.338e-7,
01783
              1.266e-7, 1.2e-7, 1.138e-7, 1.08e-7, 1.027e-7, 9.764e-8,
01785
              9.296e-8, 8.862e-8, 8.458e-8, 8.087e-8, 7.744e-8, 7.429e-8,
01786
              7.145e-8, 6.893e-8, 6.664e-8, 6.468e-8, 6.322e-8, 6.162e-8,
             6.07e-8, 5.992e-8, 5.913e-8, 5.841e-8, 5.796e-8, 5.757e-8, 5.746e-8, 5.731e-8, 5.679e-8, 5.577e-8, 5.671e-8, 5.656e-8, 5.594e-8, 5.593e-8, 5.602e-8, 5.62e-8, 5.693e-8, 5.725e-8,
01787
01788
01789
              5.858e-8, 6.037e-8, 6.249e-8, 6.535e-8, 6.899e-8, 7.356e-8,
01790
01791
              7.918e-8, 8.618e-8, 9.385e-8, 1.039e-7, 1.158e-7, 1.29e-7,
             1.437e-7, 1.65e-7, 1.871e-7, 2.121e-7, 2.427e-7, 2.773e-7, 3.247e-7, 3.677e-7, 4.037e-7, 4.776e-7, 5.101e-7, 6.214e-7,
01792
01793
              6.936e-7, 7.581e-7, 8.486e-7, 9.355e-7, 9.942e-7, 1.063e-6,
01794
01795
             1.123e-6, 1.191e-6, 1.215e-6, 1.247e-6, 1.26e-6, 1.271e-6,
              1.284e-6, 1.317e-6, 1.323e-6, 1.349e-6, 1.353e-6, 1.362e-6,
              1.344e-6, 1.329e-6, 1.336e-6, 1.327e-6, 1.325e-6, 1.359e-6,
01797
01798
              1.374e-6, 1.415e-6, 1.462e-6, 1.526e-6, 1.619e-6, 1.735e-6,
01799
             1.863e-6, 2.034e-6, 2.265e-6, 2.482e-6, 2.756e-6, 3.103e-6,
             3.466e-6, 3.832e-6, 4.378e-6, 4.913e-6, 5.651e-6, 6.311e-6, 7.169e-6, 8.057e-6, 9.253e-6, 1.047e-5, 1.212e-5, 1.36e-5,
01800
01801
```

```
1.569e-5, 1.776e-5, 2.02e-5, 2.281e-5, 2.683e-5, 2.994e-5,
              3.488e-5, 3.896e-5, 4.499e-5, 5.175e-5, 6.035e-5, 6.34e-5, 7.281e-5, 7.923e-5, 8.348e-5, 9.631e-5, 1.044e-4, 1.102e-4,
01803
01804
01805
              1.176e-4, 1.244e-4, 1.283e-4, 1.326e-4, 1.4e-4, 1.395e-4,
              1.387e-4, 1.363e-4, 1.314e-4, 1.241e-4, 1.228e-4, 1.148e-4, 1.086e-4, 1.018e-4, 8.89e-5, 8.316e-5, 7.292e-5, 6.452e-5, 5.625e-5, 5.045e-5, 4.38e-5, 3.762e-5, 3.29e-5, 2.836e-5,
01806
01807
              2.485e-5, 2.168e-5, 1.895e-5, 1.659e-5, 1.453e-5, 1.282e-5,
01809
01810
              1.132e-5, 1.001e-5, 8.836e-6, 7.804e-6, 6.922e-6, 6.116e-6,
              5.429e-6, 4.824e-6, 4.278e-6, 3.788e-6, 3.371e-6, 2.985e-6, 2.649e-6, 2.357e-6, 2.09e-6, 1.858e-6, 1.647e-6, 1.462e-6, 1.299e-6, 1.155e-6, 1.028e-6, 9.142e-7, 8.132e-7, 7.246e-7, 6.451e-7, 5.764e-7, 5.151e-7, 4.603e-7, 4.121e-7, 3.694e-7,
01811
01812
01813
01814
              3.318e-7, 2.985e-7, 2.69e-7, 2.428e-7, 2.197e-7, 1.992e-7, 1.81e-7, 1.649e-7, 1.506e-7, 1.378e-7, 1.265e-7, 1.163e-7,
01815
01816
              1.073e-7, 9.918e-8, 9.191e-8, 8.538e-8, 7.949e-8, 7.419e-8, 6.94e-8, 6.508e-8, 6.114e-8, 5.761e-8, 5.437e-8, 5.146e-8, 4.89e-8, 4.636e-8, 4.406e-8, 4.201e-8, 4.015e-8, 3.84e-8,
01817
01818
01819
              3.661e-8, 3.51e-8, 3.377e-8, 3.242e-8, 3.13e-8, 3.015e-8,
              2.918e-8, 2.83e-8, 2.758e-8, 2.707e-8, 2.656e-8, 2.619e-8,
01821
              2.609e-8, 2.615e-8, 2.63e-8, 2.675e-8, 2.745e-8, 2.842e-8,
01822
01823
              2.966e-8, 3.125e-8, 3.318e-8, 3.565e-8, 3.85e-8, 4.191e-8,
              4.59e-8, 5.059e-8, 5.607e-8, 6.239e-8, 6.958e-8, 7.796e-8,
01824
              8.773e-8, 9.88e-8, 1.114e-7, 1.258e-7, 1.422e-7, 1.61e-7, 1.822e-7, 2.06e-7, 2.337e-7, 2.645e-7, 2.996e-7, 3.393e-7, 3.843e-7, 4.363e-7, 4.935e-7, 5.607e-7, 6.363e-7, 7.242e-7, 8.23e-7, 9.411e-7, 1.071e-6, 1.232e-6, 1.402e-6, 1.6e-6, 1.82e-6,
01825
01826
01828
01829
              2.128e-6, 2.386e-6, 2.781e-6, 3.242e-6, 3.653e-6, 4.323e-6,
01830
              4.747e-6, 5.321e-6, 5.919e-6, 6.681e-6, 7.101e-6, 7.983e-6,
01831
              8.342e-6, 8.741e-6, 9.431e-6, 9.952e-6, 1.026e-5, 1.055e-5,
01832
              1.095e-5, 1.095e-5, 1.087e-5, 1.056e-5, 1.026e-5, 9.715e-6,
01833
              9.252e-6, 8.452e-6, 7.958e-6, 7.268e-6, 6.295e-6, 6.003e-6, 5e-6,
              4.591e-6, 3.983e-6, 3.479e-6, 3.058e-6, 2.667e-6, 2.293e-6,
01834
01835
              1.995e-6, 1.747e-6, 1.517e-6, 1.335e-6, 1.165e-6, 1.028e-6,
              9.007e-7, 7.956e-7, 7.015e-7, 6.192e-7, 5.491e-7, 4.859e-7, 4.297e-7, 3.799e-7, 3.38e-7, 3.002e-7, 2.659e-7, 2.366e-7, 2.103e-7, 1.861e-7, 1.655e-7, 1.469e-7, 1.309e-7, 1.162e-7, 1.032e-7, 9.198e-8, 8.181e-8, 7.294e-8, 6.516e-8, 5.787e-8,
01836
01837
01838
01840
               5.163e-8, 4.612e-8, 4.119e-8, 3.695e-8, 3.308e-8, 2.976e-8,
              2.67e-8, 2.407e-8, 2.171e-8, 1.965e-8, 1.78e-8, 1.617e-8, 1.47e-8, 1.341e-8, 1.227e-8, 1.125e-8, 1.033e-8, 9.524e-9,
01841
01842
              8.797e-9, 8.162e-9, 7.565e-9, 7.04e-9, 6.56e-9, 6.129e-9, 5.733e-9, 5.376e-9, 5.043e-9, 4.75e-9, 4.466e-9, 4.211e-9, 3.977e-9, 3.759e-9, 3.558e-9, 3.373e-9, 3.201e-9, 3.043e-9,
01843
01844
01845
              2.895e-9, 2.76e-9, 2.635e-9, 2.518e-9, 2.411e-9, 2.314e-9, 2.23e-9, 2.151e-9, 2.087e-9, 2.035e-9, 1.988e-9, 1.946e-9,
01846
01847
01848
              1.927e-9, 1.916e-9, 1.916e-9, 1.933e-9, 1.966e-9, 2.018e-9,
              2.09e-9, 2.182e-9, 2.299e-9, 2.442e-9, 2.623e-9, 2.832e-9,
01849
              3.079e-9, 3.368e-9, 3.714e-9, 4.104e-9, 4.567e-9, 5.091e-9, 5.701e-9, 6.398e-9, 7.194e-9, 8.127e-9, 9.141e-9, 1.035e-8,
01850
01851
               1.177e-8, 1.338e-8, 1.508e-8, 1.711e-8, 1.955e-8, 2.216e-8,
              2.534e-8, 2.871e-8, 3.291e-8, 3.711e-8, 4.285e-8, 4.868e-8,
01853
01854
              5.509e-8, 6.276e-8, 7.262e-8, 8.252e-8, 9.4e-8, 1.064e-7,
              1.247e-7, 1.411e-7, 1.626e-7, 1.827e-7, 2.044e-7, 2.284e-7, 2.452e-7, 2.854e-7, 3.026e-7, 3.278e-7, 3.474e-7, 3.693e-7,
01855
01856
01857
              3.93e-7, 4.104e-7, 4.22e-7, 4.439e-7, 4.545e-7, 4.778e-7,
               4.812e-7, 5.018e-7, 4.899e-7, 5.075e-7, 5.073e-7, 5.171e-
              5.131e-7, 5.25e-7, 5.617e-7, 5.846e-7, 6.239e-7, 6.696e-7,
01859
              7.398e-7, 8.073e-7, 9.15e-7, 1.009e-6, 1.116e-6, 1.264e-6
01860
01861
              1.439e-6, 1.644e-6, 1.856e-6, 2.147e-6, 2.317e-6, 2.713e-6,
              2.882e-6, 2.99e-6, 3.489e-6, 3.581e-6, 4.033e-6, 4.26e-6, 4.543e-6, 4.84e-6, 4.826e-6, 5.013e-6, 5.252e-6, 5.277e-6,
01862
01863
              5.306e-6, 5.236e-6, 5.123e-6, 5.171e-6, 4.843e-6, 4.615e-6,
              4.385e-6, 3.97e-6, 3.693e-6, 3.231e-6, 2.915e-6, 2.495e-6, 2.144e-6, 1.91e-6, 1.639e-6, 1.417e-6, 1.226e-6, 1.065e-6,
01865
01866
              9.29e-7, 8.142e-7, 7.161e-7, 6.318e-7, 5.581e-7, 4.943e-7, 4.376e-7, 3.884e-7, 3.449e-7, 3.06e-7, 2.712e-7, 2.412e-7, 2.139e-7, 1.903e-7, 1.689e-7, 1.499e-7, 1.331e-7, 1.183e-7,
01867
01868
01869
              1.05e-7, 9.362e-8, 8.306e-8, 7.403e-8, 6.578e-8, 5.853e-8,
01870
               5.216e-8, 4.632e-8, 4.127e-8, 3.678e-8, 3.279e-8, 2.923e-8,
              2.612e-8, 2.339e-8, 2.094e-8, 1.877e-8, 1.686e-8, 1.516e-8,
01872
01873
              1.366e-8, 1.234e-8, 1.114e-8, 1.012e-8, 9.182e-9, 8.362e-9,
              7.634e-9, 6.981e-9, 6.406e-9, 5.888e-9, 5.428e-9, 5.021e-9, 4.65e-9, 4.326e-9, 4.033e-9, 3.77e-9, 3.536e-9, 3.327e-9,
01874
01875
              3.141e-9, 2.974e-9, 2.825e-9, 2.697e-9, 2.584e-9, 2.488e-9,
01876
              2.406e-9, 2.34e-9, 2.292e-9, 2.259e-9, 2.244e-9, 2.243e-9,
01877
              2.272e-9, 2.31e-9, 2.378e-9, 2.454e-9, 2.618e-9, 2.672e-9,
01878
              2.831e-9, 3.05e-9, 3.225e-9, 3.425e-9, 3.677e-9, 3.968e-9, 4.221e-9, 4.639e-9, 4.96e-9, 5.359e-9, 5.649e-9, 6.23e-9, 6.716e-9, 7.218e-9, 7.746e-9, 7.988e-9, 8.627e-9, 8.999e-9,
01879
01880
01881
              9.442e-9, 9.82e-9, 1.015e-8, 1.06e-8, 1.079e-8, 1.109e-8, 1.137e-8, 1.186e-8, 1.18e-8, 1.187e-8, 1.194e-8, 1.192e-8,
01882
               1.224e-8, 1.245e-8, 1.246e-8, 1.318e-8, 1.377e-8, 1.471e-8,
01884
01885
              1.582e-8, 1.713e-8, 1.853e-8, 2.063e-8, 2.27e-8, 2.567e-8,
01886
              2.891e-8, 3.264e-8, 3.744e-8, 4.286e-8, 4.915e-8, 5.623e-8,
              6.336e-8, 7.293e-8, 8.309e-8, 9.319e-8, 1.091e-7, 1.243e-7, 1.348e-7, 1.449e-7, 1.62e-7, 1.846e-7, 1.937e-7, 2.04e-7,
01887
01888
```

```
2.179e-7, 2.298e-7, 2.433e-7, 2.439e-7, 2.464e-7, 2.611e-7,
            2.617e-7, 2.582e-7, 2.453e-7, 2.401e-7, 2.349e-7, 2.203e-7, 2.066e-7, 1.939e-7, 1.78e-7, 1.558e-7, 1.391e-7, 1.203e-7,
01890
01891
            1.048e-7, 9.464e-8, 8.306e-8, 7.239e-8, 6.317e-8, 5.52e-8,
01892
01893
            4.847e-8, 4.282e-8, 3.796e-8, 3.377e-8, 2.996e-8, 2.678e-8,
            2.4e-8, 2.134e-8, 1.904e-8, 1.705e-8, 1.523e-8, 1.35e-8,
01894
            1.204e-8, 1.07e-8, 9.408e-9, 8.476e-9, 7.47e-9, 6.679e-9,
            5.929e-9, 5.267e-9, 4.711e-9, 4.172e-9, 3.761e-9, 3.288e-9,
01896
01897
            2.929e-9, 2.609e-9, 2.315e-9, 2.042e-9, 1.844e-9, 1.64e-9,
01898
            1.47e-9, 1.31e-9, 1.176e-9, 1.049e-9, 9.377e-10, 8.462e-10,
            1.17e-3, 1.17e-3, 1.17e-3, 1.37e-10, 0.402e-10, 7.616e-10, 6.854e-10, 6.191e-10, 5.596e-10, 5.078e-10, 4.611e-10, 4.197e-10, 3.83e-10, 3.505e-10, 3.215e-10, 2.956e-10, 2.726e-10,
01899
01900
            2.521e-10, 2.338e-10, 2.173e-10, 2.026e-10, 1.895e-10, 1.777e-10,
            1.672e-10, 1.579e-10, 1.496e-10, 1.423e-10, 1.358e-10, 1.302e-10,
01902
01903
            1.254e-10, 1.216e-10, 1.187e-10, 1.163e-10, 1.147e-10, 1.145e-10,
            1.15e-10, 1.17e-10, 1.192e-10, 1.25e-10, 1.298e-10, 1.345e-10, 1.405e-10, 1.538e-10, 1.648e-10, 1.721e-10, 1.872e-10, 1.968e-10,
01904
01905
01906
            2.089e-10, 2.172e-10, 2.317e-10, 2.389e-10, 2.503e-10, 2.585e-10,
            2.686e-10, 2.8e-10, 2.895e-10, 3.019e-10, 3.037e-10, 3.076e-10,
01908
            3.146e-10, 3.198e-10, 3.332e-10, 3.397e-10, 3.54e-10, 3.667e-10,
            3.895e-10, 4.071e-10, 4.565e-10, 4.983e-10, 5.439e-10, 5.968e-10,
01909
01910
            6.676e-10, 7.456e-10, 8.405e-10, 9.478e-10, 1.064e-9, 1.218e-9,
            1.386e-9, 1.581e-9, 1.787e-9, 2.032e-9, 2.347e-9, 2.677e-9,
01911
            3.008e-9, 3.544e-9, 4.056e-9, 4.687e-9, 5.331e-9, 6.227e-9, 6.854e-9, 8.139e-9, 8.945e-9, 9.865e-9, 1.125e-8, 1.178e-8,
01912
01913
            1.364e-8, 1.436e-8, 1.54e-8, 1.672e-8, 1.793e-8, 1.906e-8,
01914
01915
            2.036e-8, 2.144e-8, 2.292e-8, 2.371e-8, 2.493e-8, 2.606e-8
01916
            2.706e-8, 2.866e-8, 3.036e-8, 3.136e-8, 3.405e-8, 3.665e-8,
            3.837e-8, 4.229e-8, 4.748e-8, 5.32e-8, 5.763e-8, 6.677e-8, 7.216e-8, 7.716e-8, 8.958e-8, 9.419e-8, 1.036e-7, 1.108e-7,
01917
01918
01919
            1.189e-7, 1.246e-7, 1.348e-7, 1.31e-7, 1.361e-7, 1.364e-7,
01920
            1.363e-7, 1.343e-7, 1.293e-7, 1.254e-7, 1.235e-7, 1.158e-
            1.107e-7, 9.961e-8, 9.011e-8, 7.91e-8, 6.916e-8, 6.338e-8,
01921
01922
            5.564e-8, 4.827e-8, 4.198e-8, 3.695e-8, 3.276e-8, 2.929e-8,
01923
            2.633e-8, 2.391e-8, 2.192e-8, 2.021e-8, 1.89e-8, 1.772e-8,
            1.667e-8, 1.603e-8, 1.547e-8, 1.537e-8, 1.492e-8, 1.515e-8,
01924
            1.479e-8, 1.45e-8, 1.513e-8, 1.495e-8, 1.529e-8, 1.565e-8, 1.564e-8, 1.553e-8, 1.569e-8, 1.584e-8, 1.57e-8, 1.538e-8,
01925
01927
            1.513e-8, 1.472e-8, 1.425e-8, 1.349e-8, 1.328e-8, 1.249e-8,
            1.17e-8, 1.077e-8, 9.514e-9, 8.614e-9, 7.46e-9, 6.621e-9,
01928
            5.775e-9, 5.006e-9, 4.308e-9, 3.747e-9, 3.24e-9, 2.84e-9, 2.481e-9, 2.184e-9, 1.923e-9, 1.71e-9, 1.504e-9, 1.334e-9, 1.187e-9, 1.053e-9, 9.367e-10, 8.306e-10, 7.419e-10, 6.63e-10,
01929
01930
01931
            5.918e-10, 5.277e-10, 4.717e-10, 4.222e-10, 3.783e-10, 3.39e-10,
            3.036e-10, 2.729e-10, 2.455e-10, 2.211e-10, 1.995e-10, 1.804e-10,
01933
01934
            1.635e-10, 1.485e-10, 1.355e-10, 1.24e-10, 1.139e-10, 1.051e-10,
           9.757e-11, 9.114e-11, 8.577e-11, 8.139e-11, 7.792e-11, 7.52e-11, 7.39e-11, 7.31le-11, 7.277e-11, 7.482e-11, 7.698e-11, 8.162e-11, 8.517e-11, 8.968e-11, 9.905e-11, 1.075e-10, 1.187e-10, 1.291e-10,
01935
01936
01937
01938
            1.426e-10, 1.573e-10, 1.734e-10, 1.905e-10, 2.097e-10, 2.28e-10,
            2.473e-10, 2.718e-10, 2.922e-10, 3.128e-10, 3.361e-10, 3.641e-10,
            3.91e-10, 4.196e-10, 4.501e-10, 4.932e-10, 5.258e-10, 5.755e-10,
01940
01941
            6.253e-10, 6.664e-10, 7.344e-10, 7.985e-10, 8.877e-10, 1.005e-9,
            1.118e-9, 1.251e-9, 1.428e-9, 1.61e-9, 1.888e-9, 2.077e-9, 2.331e-9, 2.751e-9, 3.061e-9, 3.522e-9, 3.805e-9, 4.181e-9,
01942
01943
            4.575e-9, 5.167e-9, 5.634e-9, 6.007e-9, 6.501e-9, 6.829e-9, 7.211e-9, 7.262e-9, 7.696e-9, 7.832e-9, 7.799e-9, 7.651e-9,
01944
01946
            7.304e-9, 7.15e-9, 6.977e-9, 6.603e-9, 6.209e-9, 5.69e-9,
01947
            5.432e-9, 4.764e-9, 4.189e-9, 3.64e-9, 3.203e-9, 2.848e-9,
            2.51e-9, 2.194e-9, 1.946e-9, 1.75e-9, 1.567e-9, 1.426e-9,
01948
            1.302e-9, 1.197e-9, 1.109e-9, 1.035e-9, 9.719e-10, 9.207e-10,
01949
            8.957e-10, 8.578e-10, 8.262e-10, 8.117e-10, 7.987e-10, 7.741e-10, 7.762e-10, 7.537e-10, 7.424e-10, 7.474e-10, 7.294e-10,
01950
            7.216e-10, 7.233e-10, 7.075e-10, 6.892e-10, 6.618e-10,
01952
01953
            6.208e-10, 5.689e-10, 5.55e-10, 4.984e-10, 4.6e-10, 4.078e-10,
01954
            3.879e-10, 3.459e-10, 2.982e-10, 2.626e-10, 2.329e-10, 1.988e-10,
            1.735e-10, 1.487e-10, 1.297e-10, 1.133e-10, 9.943e-11, 8.736e-11, 7.726e-11, 6.836e-11, 6.053e-11, 5.384e-11, 4.789e-11, 4.267e-11,
01955
01956
            3.804e-11, 3.398e-11, 3.034e-11, 2.71e-11, 2.425e-11, 2.173e-11,
01957
            1.95e-11, 1.752e-11, 1.574e-11, 1.418e-11, 1.278e-11, 1.154e-11,
            1.044e-11, 9.463e-12, 8.602e-12, 7.841e-12, 7.171e-12, 6.584e-12,
01959
01960
            6.073e-12, 5.631e-12, 5.254e-12, 4.937e-12, 4.679e-12, 4.476e-12,
01961
            4.328e-12, 4.233e-12, 4.194e-12, 4.211e-12, 4.286e-12, 4.424e-12,
            4.628e-12, 4.906e-12, 5.262e-12, 5.708e-12, 6.254e-12, 6.914e-12,
01962
            7.714e-12, 8.677e-12, 9.747e-12, 1.101e-11, 1.256e-11, 1.409e-11,
01963
            1.597e-11, 1.807e-11, 2.034e-11, 2.316e-11, 2.622e-11, 2.962e-11,
01964
01965
            3.369e-11, 3.819e-11, 4.329e-11, 4.932e-11, 5.589e-11, 6.364e-11,
01966
            7.284e-11, 8.236e-11, 9.447e-11, 1.078e-10, 1.229e-10, 1.417e-10,
01967
            1.614e-10, 1.843e-10, 2.107e-10, 2.406e-10, 2.728e-10, 3.195e-10,
            3.595e-10, 4.153e-10, 4.736e-10, 5.41e-10, 6.088e-10, 6.769e-10, 7.691e-10, 8.545e-10, 9.621e-10, 1.047e-9, 1.161e-9, 1.296e-9,
01968
01969
            1.424e-9, 1.576e-9, 1.739e-9, 1.893e-9, 2.08e-9, 2.336e-9,
01971
            2.604e-9, 2.76e-9, 3.001e-9, 3.365e-9, 3.55e-9, 3.895e-9,
01972
            4.183e-9, 4.614e-9, 4.846e-9, 5.068e-9, 5.427e-9, 5.541e-9,
01973
            5.864e-9, 5.997e-9, 5.997e-9, 6.061e-9, 5.944e-9, 5.855e-9,
            5.661e-9, 5.523e-9, 5.374e-9, 4.94e-9, 4.688e-9, 4.17e-9, 3.913e-9, 3.423e-9, 2.997e-9, 2.598e-9, 2.253e-9, 1.946e-9,
01974
01975
```

```
1.71e-9, 1.507e-9, 1.336e-9, 1.19e-9, 1.068e-9, 9.623e-10,
             8.772e-10, 8.007e-10, 7.42e-10, 6.884e-10, 6.483e-10, 6.162e-10, 5.922e-10, 5.688e-10, 5.654e-10, 5.637e-10, 5.701e-10, 5.781e-10,
01977
01978
01979
             5.874e-10, 6.268e-10, 6.357e-10, 6.525e-10, 7.137e-10, 7.441e-10,
01980
             8.024e-10, 8.485e-10, 9.143e-10, 9.536e-10, 9.717e-10, 1.018e-9,
             1.042e-9, 1.054e-9, 1.092e-9, 1.079e-9, 1.064e-9, 1.043e-9,
01981
             1.02e-9, 9.687e-10, 9.273e-10, 9.208e-10, 9.068e-10, 7.687e-10,
             7.385e-10, 6.595e-10, 5.87e-10, 5.144e-10, 4.417e-10, 3.804e-10,
01983
01984
             3.301e-10, 2.866e-10, 2.509e-10, 2.202e-10, 1.947e-10, 1.719e-10,
01985
             1.525e-10, 1.361e-10, 1.21e-10, 1.084e-10, 9.8e-11, 8.801e-11,
             7.954e-11, 7.124e-11, 6.335e-11, 5.76e-11, 5.132e-11, 4.601e-11,
01986
             4.096e-11, 3.657e-11, 3.25e-11, 2.909e-11, 2.587e-11, 2.297e-11,
01987
             2.05e-11, 1.828e-11, 1.632e-11, 1.462e-11, 1.314e-11, 1.185e-11, 1.073e-11, 9.76e-12, 8.922e-12, 8.206e-12, 7.602e-12, 7.1e-12,
01988
01989
01990
             6.694e-12, 6.378e-12, 6.149e-12, 6.004e-12, 5.941e-12, 5.962e-12,
01991
             6.069e-12, 6.265e-12, 6.551e-12, 6.935e-12, 7.457e-12, 8.074e-12,
            8.811e-12, 9.852e-12, 1.086e-11, 1.207e-11, 1.361e-11, 1.553e-11, 1.737e-11, 1.93e-11, 2.175e-11, 2.41e-11, 2.706e-11, 3.023e-11, 3.313e-11, 3.657e-11, 4.118e-11, 4.569e-11, 5.025e-11, 5.66e-11,
01992
01993
01995
             6.231e-11, 6.881e-11, 7.996e-11, 8.526e-11, 9.694e-11, 1.106e-10,
             1.222e-10, 1.355e-10, 1.525e-10, 1.775e-10, 1.924e-10, 2.181e-10,
01996
01997
             2.379e-10, 2.662e-10, 2.907e-10, 3.154e-10, 3.366e-10, 3.579e-10,
            3.858e-10, 4.046e-10, 4.196e-10, 4.166e-10, 4.457e-10, 4.466e-10, 4.404e-10, 4.337e-10, 4.15e-10, 4.083e-10, 3.91e-10, 3.723e-10, 3.514e-10, 3.303e-10, 2.847e-10, 2.546e-10, 2.23e-10, 1.994e-10,
01998
01999
02000
02001
             1.733e-10, 1.488e-10, 1.297e-10, 1.144e-10, 1.004e-10, 8.741e-11,
             7.928e-11, 7.034e-11, 6.323e-11, 5.754e-11, 5.25e-11, 4.85e-11,
02002
02003
             4.502e-11, 4.286e-11, 4.028e-11, 3.899e-11, 3.824e-11, 3.761e-11,
02004
             3.804 e^{-11}, \ 3.839 e^{-11}, \ 3.845 e^{-11}, \ 4.244 e^{-11}, \ 4.382 e^{-11}, \ 4.582 e^{-11},
02005
             4.847e-11, 5.209e-11, 5.384e-11, 5.887e-11, 6.371e-11, 6.737e-11,
02006
             7.168e-11, 7.415e-11, 7.827e-11, 8.037e-11, 8.12e-11, 8.071e-11,
02007
             8.008e-11, 7.851e-11, 7.544e-11, 7.377e-11, 7.173e-11, 6.801e-11,
             6.267e-11, 5.727e-11, 5.288e-11, 4.853e-11, 4.082e-11, 3.645e-11,
02008
02009
             3.136e-11, 2.672e-11, 2.304e-11, 1.986e-11, 1.725e-11, 1.503e-11,
             1.315e-11, 1.153e-11, 1.014e-11, 8.942e-12, 7.901e-12, 6.993e-12, 6.199e-12, 5.502e-12, 4.89e-12, 4.351e-12, 3.878e-12, 3.461e-12,
02010
02011
             3.094e-12, 2.771e-12, 2.488e-12, 2.241e-12, 2.025e-12, 1.838e-12,
02012
             1.677e-12, 1.541e-12, 1.427e-12, 1.335e-12, 1.262e-12, 1.209e-12,
02014
             1.176e-12, 1.161e-12, 1.165e-12, 1.189e-12, 1.234e-12, 1.3e-12,
             1.389e-12, 1.503e-12, 1.644e-12, 1.814e-12, 2.017e-12, 2.255e-12,
02015
            2.534e-12, 2.858e-12, 3.231e-12, 3.661e-12, 4.153e-12, 4.717e-12, 5.36e-12, 6.094e-12, 6.93e-12, 7.882e-12, 8.966e-12, 1.02e-11, 1.162e-11, 1.324e-11, 1.51e-11, 1.72e-11, 1.965e-11, 2.237e-11, 2.56e-11, 2.927e-11, 3.371e-11, 3.842e-11, 4.429e-11, 5.139e-11,
02016
02017
02018
             5.798e-11, 6.697e-11, 7.626e-11, 8.647e-11, 1.022e-10, 1.136e-10,
02020
02021
             1.3e-10, 1.481e-10, 1.672e-10, 1.871e-10, 2.126e-10, 2.357e-10,
02022
             2.583e-10, 2.997e-10, 3.289e-10, 3.702e-10, 4.012e-10, 4.319e-10,
02023
             4.527e-10, 5.001e-10, 5.448e-10, 5.611e-10, 5.76e-10, 5.965e-10,
02024
             6.079e-10, 6.207e-10, 6.276e-10, 6.222e-10, 6.137e-10, 6e-10,
             5.814e-10, 5.393e-10, 5.35e-10, 4.947e-10, 4.629e-10, 4.117e-10,
02025
             3.712e-10, 3.372e-10, 2.923e-10, 2.55e-10, 2.232e-10, 1.929e-10,
             1.679e-10, 1.46e-10, 1.289e-10, 1.13e-10, 9.953e-11, 8.763e-11,
02027
02028
             7.76e-11, 6.9e-11, 6.16e-11, 5.525e-11, 4.958e-11, 4.489e-11,
             4.072e-11, 3.728e-11, 3.438e-11, 3.205e-11, 3.006e-11, 2.848e-11, 2.766e-11, 2.688e-11, 2.664e-11, 2.67e-11, 2.696e-11, 2.786e-11,
02029
02030
             2.861e-11, 3.009e-11, 3.178e-11, 3.389e-11, 3.587e-11, 3.819e-11, 4.054e-11, 4.417e-11, 4.703e-11, 5.137e-11, 5.46e-11, 6.055e-11,
02031
             6.333e-11, 6.773e-11, 7.219e-11, 7.717e-11, 8.131e-11, 8.491e-11,
02033
             8.574e-11, 9.01e-11, 9.017e-11, 8.999e-11, 8.959e-11, 8.838e-11,
02034
02035
             8.579e-11, 8.162e-11, 8.098e-11, 7.472e-11, 7.108e-11, 6.559e-11,
            5.994e-11, 5.172e-11, 4.424e-11, 3.951e-11, 3.34e-11, 2.902e-11, 2.541e-11, 2.215e-11, 1.945e-11, 1.716e-11, 1.503e-11, 1.339e-11, 1.185e-11, 1.05e-11, 9.336e-12, 8.307e-12, 7.312e-12, 6.55e-12,
02036
02037
             5.836e-12, 5.178e-12, 4.6e-12, 4.086e-12, 3.639e-12, 3.247e-12,
02039
02040
             2.904e-12, 2.604e-12, 2.341e-12, 2.112e-12, 1.914e-12, 1.744e-12,
02041
             1.598e-12, 1.476e-12, 1.374e-12, 1.293e-12, 1.23e-12, 1.185e-12,
             1.158e-12, 1.147e-12, 1.154e-12, 1.177e-12, 1.219e-12, 1.28e-12, 1.36e-12, 1.463e-12, 1.591e-12, 1.75e-12, 1.94e-12, 2.156e-12, 2.43e-12, 2.748e-12, 3.052e-12, 3.533e-12, 3.967e-12, 4.471e-12,
02042
02043
02044
             5.041e-12, 5.86e-12, 6.664e-12, 7.522e-12, 8.342e-12, 9.412e-12,
02046
             1.072e-11, 1.213e-11, 1.343e-11, 1.496e-11, 1.664e-11, 1.822e-11,
02047
             2.029e-11, 2.233e-11, 2.457e-11, 2.709e-11, 2.928e-11, 3.115e-11,
02048
             3.356e-11, 3.592e-11, 3.818e-11, 3.936e-11, 4.061e-11, 4.149e-11,
             4.299e-11, 4.223e-11, 4.251e-11, 4.287e-11, 4.177e-11, 4.094e-11, 3.942e-11, 3.772e-11, 3.614e-11, 3.394e-11, 3.222e-11, 2.791e-11, 2.665e-11, 2.309e-11, 2.032e-11, 1.74e-11, 1.535e-11, 1.323e-11, 1.151e-11, 9.803e-12, 8.65e-12, 7.54e-12, 6.619e-12, 5.832e-12,
02049
02050
02051
02052
02053
             5.113e-12, 4.503e-12, 3.975e-12, 3.52e-12, 3.112e-12, 2.797e-12,
02054
             2.5e-12, 2.24e-12, 2.013e-12, 1.819e-12, 1.653e-12, 1.513e-12,
             1.395e-12, 1.299e-12, 1.225e-12, 1.168e-12, 1.124e-12, 1.148e-12,
02055
             1.107e-12, 1.128e-12, 1.169e-12, 1.233e-12, 1.307e-12, 1.359e-12, 1.543e-12, 1.686e-12, 1.794e-12, 2.028e-12, 2.21e-12, 2.441e-12,
02056
             2.653e-12, 2.828e-12, 3.093e-12, 3.28e-12, 3.551e-12, 3.677e-12,
02058
02059
             3.803e-12, 3.844e-12, 4.068e-12, 4.093e-12, 4.002e-12, 3.904e-12,
02060
             3.624e-12, 3.633e-12, 3.622e-12, 3.443e-12, 3.184e-12, 2.934e-12,
             2.476e-12, 2.212e-12, 1.867e-12, 1.594e-12, 1.37e-12, 1.192e-12, 1.045e-12, 9.211e-13, 8.17e-13, 7.29e-13, 6.55e-13, 5.929e-13,
02061
02062
```

```
5.415e-13, 4.995e-13, 4.661e-13, 4.406e-13, 4.225e-13, 4.116e-13,
                  4.075e-13, 4.102e-13, 4.198e-13, 4.365e-13, 4.606e-13, 4.925e-13,
02064
02065
                  5.326e-13, 5.818e-13, 6.407e-13, 7.104e-13, 7.92e-13, 8.868e-13,
02066
                  9.964e-13, 1.123e-12, 1.268e-12, 1.434e-12, 1.626e-12, 1.848e-12,
02067
                   2.107e-12, \ 2.422e-12, \ 2.772e-12, \ 3.145e-12, \ 3.704e-12, \ 4.27e-12, \\
                  4.721e-12, 5.361e-12, 6.083e-12, 7.095e-12, 7.968e-12, 9.228e-12,
02068
                  1.048e-11, 1.187e-11, 1.336e-11, 1.577e-11, 1.772e-11, 2.017e-11,
                  2.25e-11, 2.63e-11, 2.911e-11, 3.356e-11, 3.82e-11, 4.173e-11,
02070
02071
                  4.811e-11, 5.254e-11, 5.839e-11, 6.187e-11, 6.805e-11, 7.118e-11,
02072
                  7.369e-11, 7.664e-11, 7.794e-11, 7.947e-11, 8.036e-11, 7.954e-11,
                  7.849e-11, 7.518e-11, 7.462e-11, 6.926e-11, 6.531e-11, 6.197e-11, 5.421e-11, 4.777e-11, 4.111e-11, 3.679e-11, 3.166e-11, 2.786e-11, 2.436e-11, 2.144e-11, 1.859e-11, 1.628e-11, 1.414e-11, 1.237e-11,
02073
02074
02075
02076
                  1.093e-11, 9.558e-12
02077
02078
              static double h2o260[2001] = \{ .2752, .2732, .2749, .2676, .2667, .2545, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .2749, .274
02079
                  .2497, .2327, .2218, .2036, .1825, .1694, .1497, .1353, .121, .1014, .09405, .07848, .07195, .06246, .05306, .04853, .04138, .03735, .03171, .02785, .02431, .02111, .01845, .0164, .01405,
02080
02082
                   .01255, .01098, .009797, .008646, .007779, .006898, .006099,
02083
02084
                   .005453, .004909, .004413, .003959, .003581, .003199, .002871,
                  .002583, .00233, .002086, .001874, .001684, .001512, .001361, .001225, .0011, 9.89e-4, 8.916e-4, 8.039e-4, 7.256e-4, 6.545e-4,
02085
02086
02087
                  5.918e-4, 5.359e-4, 4.867e-4, 4.426e-4, 4.033e-4, 3.682e-4,
02088
                  3.366e-4, 3.085e-4, 2.833e-4, 2.605e-4, 2.403e-4, 2.221e-4,
02089
                  2.055e-4, 1.908e-4, 1.774e-4, 1.653e-4, 1.544e-4, 1.443e-4,
02090
                  1.351e-4, 1.267e-4, 1.19e-4, 1.119e-4, 1.053e-4, 9.922e-5,
                  9.355e-5, 8.831e-5, 8.339e-5, 7.878e-5, 7.449e-5, 7.043e-5, 6.664e-5, 6.307e-5, 5.969e-5, 5.654e-5, 5.357e-5, 5.075e-5,
02091
02092
02093
                  4.81e-5, 4.56e-5, 4.322e-5, 4.102e-5, 3.892e-5, 3.696e-5,
                  3.511e-5, 3.339e-5, 3.177e-5, 3.026e-5, 2.886e-5, 2.756e-5,
                  2.636e-5, 2.527e-5, 2.427e-5, 2.337e-5, 2.257e-5, 2.185e-5
02095
02096
                  2.127e-5, 2.08e-5, 2.041e-5, 2.013e-5, 2e-5, 1.997e-5, 2.009e-5,
                  2.031e-5, 2.068e-5, 2.124e-5, 2.189e-5, 2.267e-5, 2.364e-5, 2.463e-5, 2.618e-5, 2.774e-5, 2.937e-5, 3.144e-5, 3.359e-5,
02097
02098
                  3.695e-5, 4.002e-5, 4.374e-5, 4.947e-5, 5.431e-5, 6.281e-5,
02099
                   7.169e-5, 8.157e-5, 9.728e-5, 1.079e-4, 1.337e-4, 1.442e-4,
02101
                  1.683e-4, 1.879e-4, 2.223e-4, 2.425e-4, 2.838e-4, 3.143e-4,
                  3.527e-4, 4.012e-4, 4.237e-4, 4.747e-4, 5.057e-4, 5.409e-4,
02102
02103
                  5.734e-4, 5.944e-4, 6.077e-4, 6.175e-4, 6.238e-4, 6.226e-4,
                  6.248e-4, 6.192e-4, 6.098e-4, 5.818e-4, 5.709e-4, 5.465e-4,
02104
                  5.043e-4, 4.699e-4, 4.294e-4, 3.984e-4, 3.672e-4, 3.152e-4, 2.883e-4, 2.503e-4, 2.211e-4, 1.92e-4, 1.714e-4, 1.485e-4,
02105
                  1.358e-4, 1.156e-4, 1.021e-4, 8.887e-5, 7.842e-5, 7.12e-5,
02107
02108
                  6.186e-5, 5.73e-5, 4.792e-5, 4.364e-5, 3.72e-5, 3.28e-5,
02109
                  2.946e-5, 2.591e-5, 2.261e-5, 2.048e-5, 1.813e-5, 1.63e-5,
02110
                  1.447e-5, 1.282e-5, 1.167e-5, 1.041e-5, 9.449e-6, 8.51e-6,
                  7.596e-6, 6.961e-6, 6.272e-6, 5.728e-6, 5.198e-6, 4.667e-6,
02111
02112
                  4.288e-6, 3.897e-6, 3.551e-6, 3.235e-6, 2.952e-6, 2.688e-6,
                  2.449e-6, 2.241e-6, 2.05e-6, 1.879e-6, 1.722e-6, 1.582e-6,
                  1.456e-6, 1.339e-6, 1.236e-6, 1.144e-6, 1.06e-6, 9.83e-7,
02114
02115
                  9.149e-7, 8.535e-7, 7.973e-7, 7.466e-7, 6.999e-7, 6.574e-7,
                  6.18e-7, 5.821e-7, 5.487e-7, 5.18e-7, 4.896e-7, 4.631e-7, 4.386e-7, 4.16e-7, 3.945e-7, 3.748e-7, 3.562e-7, 3.385e-7, 3.222e-7, 3.068e-7, 2.922e-7, 2.788e-7, 2.659e-7, 2.539e-7,
02116
02117
02118
                  2.425e-7, 2.318e-7, 2.219e-7, 2.127e-7, 2.039e-7, 1.958e-7,
                  1.885e-7, 1.818e-7, 1.758e-7, 1.711e-7, 1.662e-7, 1.63e-7,
02120
                 1.605e-7, 1.60e-7, 1.75e-7, 1.71e-7, 1.602e-7, 1.63e-7, 1.63e-7, 1.505e-7, 1.505e-7, 1.505e-7, 1.505e-7, 1.505e-7, 1.505e-7, 1.505e-7, 1.485e-7, 1.485e-7, 1.485e-7, 1.485e-7, 1.445e-7, 1.446e-7, 1.436e-7, 1.425e-7, 1.445e-7, 1.477e-7, 1.515e-7, 1.567e-7, 1.634e-7, 1.712e-7, 1.802e-7,
02121
02122
02123
02124
                  1.914e-7, 2.024e-7, 2.159e-7, 2.295e-7, 2.461e-7, 2.621e-7,
                  2.868e-7, 3.102e-7, 3.394e-7, 3.784e-7, 4.223e-7, 4.864e-7, 5.501e-7, 6.039e-7, 7.193e-7, 7.728e-7, 9.514e-7, 1.073e-6,
02126
02127
                  1.18e-6, 1.333e-6, 1.472e-6, 1.566e-6, 1.677e-6, 1.784e-6, 1.904e-6, 1.953e-6, 2.02e-6, 2.074e-6, 2.128e-6, 2.162e-6,
02128
02129
                  2.219e-6, 2.221e-6, 2.249e-6, 2.239e-6, 2.235e-6, 2.185e-6,
02130
                  2.141e-6, 2.124e-6, 2.09e-6, 2.068e-6, 2.1e-6, 2.104e-6,
02131
                  2.142e-6, 2.181e-6, 2.257e-6, 2.362e-6, 2.5e-6, 2.664e-6,
                  2.884e-6, 3.189e-6, 3.48e-6, 3.847e-6, 4.313e-6, 4.79e-6, 5.25e-6, 5.989e-6, 6.692e-6, 7.668e-6, 8.52e-6, 9.606e-6,
02133
02134
                  1.073e-5, 1.225e-5, 1.377e-5, 1.582e-5, 1.761e-5, 2.029e-5, 2.284e-5, 2.602e-5, 2.94e-5, 3.483e-5, 3.928e-5, 4.618e-5, 5.24e-5, 6.132e-5, 7.183e-5, 8.521e-5, 9.111e-5, 1.07e-4,
02135
02136
02137
                  1.184e-4, 1.264e-4, 1.475e-4, 1.612e-4, 1.704e-4, 1.818e-4,
02138
                  1.924e-4, 1.994e-4, 2.061e-4, 2.18e-4, 2.187e-4, 2.2e-4,
02139
                  2.196e-4, 2.131e-4, 2.015e-4, 1.988e-4, 1.847e-4, 1.729e-4, 1.597e-4, 1.373e-4, 1.262e-4, 1.087e-4, 9.439e-5, 8.061e-5,
02140
02141
                  1.37/e-4, 1.37/e-4, 1.202e-4, 1.00/e-4, 9.439e-5, 8.001e-5, 7.093e-5, 6.049e-5, 5.12e-5, 4.435e-5, 3.817e-5, 3.34e-5, 2.927e-5, 2.573e-5, 2.291e-5, 2.04e-5, 1.827e-5, 1.827e-5, 1.636e-5, 1.463e-5, 1.309e-5, 1.17e-5, 1.047e-5, 9.315e-6, 8.328e-6,
02142
02143
                   7.458e-6, 6.665e-6, 5.94e-6, 5.316e-6, 4.752e-6, 4.252e-6,
02145
02146
                  3.825e-6, 3.421e-6, 3.064e-6, 2.746e-6, 2.465e-6, 2.216e-6,
                  1.99e-6, 1.79e-6, 1.609e-6, 1.449e-6, 1.306e-6, 1.177e-6, 1.063e-6, 9.607e-7, 8.672e-7, 7.855e-7, 7.118e-7, 6.46e-7, 5.871e-7, 5.34e-7, 4.868e-7, 4.447e-7, 4.068e-7, 3.729e-7,
02147
02148
02149
```

```
3.423e-7, 3.151e-7, 2.905e-7, 2.686e-7, 2.484e-7, 2.306e-7,
            2.142e-7, 1.995e-7, 1.86e-7, 1.738e-7, 1.626e-7, 1.522e-7, 1.427e-7, 1.338e-7, 1.258e-7, 1.183e-7, 1.116e-7, 1.056e-7,
02151
02152
            9.972e-8, 9.46e-8, 9.007e-8, 8.592e-8, 8.195e-8, 7.816e-8,
02153
            7.483e-8, 7.193e-8, 6.892e-8, 6.642e-8, 6.386e-8, 6.154e-8, 5.949e-8, 5.764e-8, 5.622e-8, 5.479e-8, 5.364e-8, 5.301e-8,
02154
02155
            5.267e-8, 5.263e-8, 5.313e-8, 5.41e-8, 5.55e-8, 5.745e-8,
             6.003e-8, 6.311e-8, 6.713e-8, 7.173e-8, 7.724e-8, 8.368e-8,
02157
02158
            9.121e-8, 9.986e-8, 1.097e-7, 1.209e-7, 1.338e-7, 1.486e-7,
02159
            1.651e-7, 1.837e-7, 2.048e-7, 2.289e-7, 2.557e-7, 2.857e-7,
            3.195e-7, 3.587e-7, 4.015e-7, 4.497e-7, 5.049e-7, 5.665e-7,
02160
            6.366e-7, 7.121e-7, 7.996e-7, 8.946e-7, 1.002e-6, 1.117e-6,
02161
            1.262e-6, 1.416e-6, 1.611e-6, 1.807e-6, 2.056e-6, 2.351e-6,
02162
            2.769e-6, 3.138e-6, 3.699e-6, 4.386e-6, 5.041e-6, 6.074e-6,
02163
02164
            6.812e-6, 7.79e-6, 8.855e-6, 1.014e-5, 1.095e-5, 1.245e-5,
            1.316e-5, 1.39e-5, 1.504e-5, 1.583e-5, 1.617e-5, 1.652e-5, 1.713e-5, 1.724e-5, 1.715e-5, 1.668e-5, 1.629e-5, 1.552e-5, 1.478e-5, 1.34e-5, 1.245e-5, 1.121e-5, 9.575e-6, 8.956e-6,
02165
02166
02167
             7.345e-6, 6.597e-6, 5.612e-6, 4.818e-6, 4.165e-6, 3.579e-6,
02168
            3.041e-6, 2.623e-6, 2.29e-6, 1.984e-6, 1.748e-6, 1.534e-6,
02169
            1.369e-6, 1.219e-6, 1.092e-6, 9.8e-7, 8.762e-7, 7.896e-7, 7.104e-7, 6.364e-7, 5.691e-7, 5.107e-7, 4.575e-7, 4.09e-7,
02170
02171
            3.667e-7, 3.287e-7, 2.931e-7, 2.633e-7, 2.356e-7, 2.111e-7,
02172
            1.895e-7, 1.697e-7, 1.525e-7, 1.369e-7, 1.233e-7, 1.114e-7, 9.988e-8, 9.004e-8, 8.149e-8, 7.352e-8, 6.662e-8, 6.03e-8,
02173
02174
02175
            5.479e-8, 4.974e-8, 4.532e-8, 4.129e-8, 3.781e-8, 3.462e-8,
            3.176e-8, 2.919e-8, 2.687e-8, 2.481e-8, 2.292e-8, 2.119e-8,
02176
02177
            1.967e-8, 1.828e-8, 1.706e-8, 1.589e-8, 1.487e-8, 1.393e-8,
02178
            1.307e-8, 1.228e-8, 1.156e-8, 1.089e-8, 1.028e-8, 9.696e-9,
02179
            9.159e-9, 8.658e-9, 8.187e-9, 7.746e-9, 7.34e-9, 6.953e-9,
            6.594e-9, 6.259e-9, 5.948e-9, 5.66e-9, 5.386e-9, 5.135e-9,
02180
02181
             4.903e-9, 4.703e-9, 4.515e-9, 4.362e-9, 4.233e-9, 4.117e-9,
            4.017e-9, 3.962e-9, 3.924e-9, 3.905e-9, 3.922e-9, 3.967e-9,
02182
02183
            4.046e-9, 4.165e-9, 4.32e-9, 4.522e-9, 4.769e-9, 5.083e-9,
            5.443e-9, 5.872e-9, 6.366e-9, 6.949e-9, 7.601e-9, 8.371e-9, 9.22e-9, 1.02e-8, 1.129e-8, 1.251e-8, 1.393e-8, 1.542e-8, 1.72e-8, 1.926e-8, 2.152e-8, 2.392e-8, 2.678e-8, 3.028e-8,
02184
02185
02186
            3.39e-8, 3.836e-8, 4.309e-8, 4.9e-8, 5.481e-8, 6.252e-8,
02188
            7.039e-8, 7.883e-8, 8.849e-8, 1.012e-7, 1.142e-7, 1.3e-7,
            1.475e-7, 1.732e-7, 1.978e-7, 2.304e-7, 2.631e-7, 2.988e-7,
02189
            3.392e-7, 3.69e-7, 4.355e-7, 4.672e-7, 5.11e-7, 5.461e-7, 5.828e-7, 6.233e-7, 6.509e-7, 6.672e-7, 6.969e-7, 7.104e-7,
02190
02191
            7.439e-7, 7.463e-7, 7.708e-7, 7.466e-7, 7.668e-7, 7.549e-7, 7.586e-7, 7.384e-7, 7.439e-7, 7.785e-7, 7.915e-7, 8.31e-7,
02192
02193
            8.745e-7, 9.558e-7, 1.038e-6, 1.173e-6, 1.304e-6, 1.452e-6,
02194
02195
            1.671e-6, 1.931e-6, 2.239e-6, 2.578e-6, 3.032e-6,
                                                                             3.334e-6.
02196
            3.98e-6, 4.3e-6, 4.518e-6, 5.321e-6, 5.508e-6, 6.211e-6, 6.59e-6,
            7.046e-6, 7.555e-6, 7.558e-6, 7.875e-6, 8.319e-6, 8.433e-6, 8.59e-6, 8.503e-6, 8.304e-6, 8.336e-6, 7.739e-6, 7.301e-6,
02197
02198
            6.827e-6, 6.078e-6, 5.551e-6, 4.762e-6, 4.224e-6, 3.538e-6,
02199
            2.984e-6, 2.619e-6, 2.227e-6, 1.923e-6, 1.669e-6, 1.462e-6,
            1.294e-6, 1.155e-6, 1.033e-6, 9.231e-7, 8.238e-7, 7.36e-7,
02201
02202
            6.564e-7, 5.869e-7, 5.236e-7, 4.673e-7, 4.174e-7, 3.736e-7,
            3.33e-7, 2.976e-7, 2.657e-7, 2.367e-7, 2.106e-7, 1.877e-7, 1.671e-7, 1.494e-7, 1.332e-7, 1.192e-7, 1.065e-7, 9.558e-8,
02203
02204
            8.586e-8, 7.717e-8, 6.958e-8, 6.278e-8, 5.666e-8, 5.121e-8,
02205
            4.647e-8, 4.213e-8, 3.815e-8, 3.459e-8, 3.146e-8, 2.862e-8,
            2.604e-8, 2.375e-8, 2.162e-8, 1.981e-8, 1.817e-8, 1.67e-8,
02207
            1.537e-8, 1.417e-8, 1.31e-8, 1.215e-8, 1.128e-8, 1.05e-8,
02208
            9.793e-9, 9.158e-9, 8.586e-9, 8.068e-9, 7.595e-9, 7.166e-9, 6.778e-9, 6.427e-9, 6.108e-9, 5.826e-9, 5.571e-9, 5.347e-9,
02209
02210
            5.144e-9, 4.968e-9, 4.822e-9, 4.692e-9, 4.589e-9, 4.506e-9,
02211
            4.467e-9, 4.44e-9, 4.466e-9, 4.515e-9, 4.718e-9, 4.729e-9,
            4.937e-9, 5.249e-9, 5.466e-9, 5.713e-9, 6.03e-9, 6.436e-9, 6.741e-9, 7.33e-9, 7.787e-9, 8.414e-9, 8.908e-9, 9.868e-9,
02213
02214
02215
            1.069e-8, 1.158e-8, 1.253e-8, 1.3e-8, 1.409e-8, 1.47e-8,
02216
            1.548e-8, 1.612e-8, 1.666e-8, 1.736e-8, 1.763e-8, 1.812e-8, 1.852e-8, 1.923e-8, 1.897e-8, 1.893e-8, 1.888e-8, 1.868e-8,
02217
            1.895e-8, 1.899e-8, 1.876e-8, 1.96e-8, 2.02e-8, 2.121e-8,
02218
            2.239e-8, 2.379e-8, 2.526e-8, 2.766e-8, 2.994e-8, 3.332e-8,
            3.703e-8, 4.158e-8, 4.774e-8, 5.499e-8, 6.355e-8, 7.349e-8,
02220
02221
            8.414e-8, 9.846e-8, 1.143e-7, 1.307e-7, 1.562e-7, 1.817e-7,
            2.011e-7, 2.192e-7, 2.485e-7, 2.867e-7, 3.035e-7, 3.223e-7, 3.43e-7, 3.617e-7, 3.793e-7, 3.793e-7, 3.839e-7, 4.081e-7, 4.117e-7, 4.085e-7, 3.92e-7, 3.851e-7, 3.754e-7, 3.49e-7, 3.229e-7, 2.978e-7, 2.691e-7, 2.312e-7, 2.029e-7, 1.721e-7,
02222
02223
02224
02225
            1.472e-7, 1.308e-7, 1.132e-7, 9.736e-8, 8.458e-8, 7.402e-8,
02226
02227
            6.534e-8, 5.811e-8, 5.235e-8, 4.762e-8, 4.293e-8, 3.896e-8,
02228
            3.526e-8, 3.165e-8, 2.833e-8, 2.551e-8, 2.288e-8, 2.036e-8,
            1.82e-8, 1.626e-8, 1.438e-8, 1.299e-8, 1.149e-8, 1.03e-8,
02229
            9.148e-9, 8.122e-9, 7.264e-9, 6.425e-9, 5.777e-9, 5.06e-9, 4.502e-9, 4.013e-9, 3.567e-9, 3.145e-9, 2.864e-9, 2.553e-9,
02230
            2.311e-9, 2.087e-9, 1.886e-9, 1.716e-9, 1.556e-9, 1.432e-9,
02232
02233
            1.311e-9, 1.202e-9, 1.104e-9, 1.013e-9, 9.293e-10, 8.493e-10,
02234
            7.79e-10, 7.185e-10, 6.642e-10, 6.141e-10, 5.684e-10, 5.346e-10,
            5.032e-10, 4.725e-10, 4.439e-10, 4.176e-10, 3.93e-10, 3.714e-10, 3.515e-10, 3.332e-10, 3.167e-10, 3.02e-10, 2.887e-10, 2.769e-10,
02235
02236
```

```
2.665e-10, 2.578e-10, 2.503e-10, 2.436e-10, 2.377e-10, 2.342e-10,
                2.305e-10, 2.296e-10, 2.278e-10, 2.321e-10, 2.355e-10, 2.402e-10, 2.478e-10, 2.67e-10, 2.848e-10, 2.982e-10, 3.263e-10, 3.438e-10,
02238
02239
02240
                3.649e-10, 3.829e-10, 4.115e-10, 4.264e-10, 4.473e-10, 4.63e-10,
02241
                4.808e-10, 4.995e-10, 5.142e-10, 5.313e-10, 5.318e-10, 5.358e-10, 5.452e-10, 5.507e-10, 5.698e-10, 5.782e-10, 5.983e-10, 6.164e-10,
02242
                 6.532e-10, 6.811e-10, 7.624e-10, 8.302e-10, 9.067e-10,
02243
                 1.104e-9, 1.221e-9, 1.361e-9, 1.516e-9, 1.675e-9, 1.883e-9,
02244
02245
                2.101e-9, 2.349e-9, 2.614e-9, 2.92e-9, 3.305e-9, 3.724e-9,
                4.142e-9, 4.887e-9, 5.614e-9, 6.506e-9, 7.463e-9, 8.817e-9, 9.849e-9, 1.187e-8, 1.321e-8, 1.474e-8, 1.698e-8, 1.794e-8,
02246
02247
                2.09e-8, 2.211e-8, 2.362e-8, 2.556e-8, 2.729e-8, 2.88e-8,
02248
02249
                 3.046e-8, 3.167e-8, 3.367e-8, 3.457e-8, 3.59e-8, 3.711e-8,
                3.826e-8, 4.001e-8, 4.211e-8, 4.315e-8, 4.661e-8, 5.01e-8, 5.249e-8, 5.84e-8, 6.628e-8, 7.512e-8, 8.253e-8, 9.722e-8,
02250
02251
                1.067e-7, 1.153e-7, 1.347e-7, 1.428e-7, 1.577e-7, 1.694e-7, 1.833e-7, 1.938e-7, 2.108e-7, 2.059e-7, 2.157e-7, 2.185e-7,
02252
02253
                2.208e-7, 2.182e-7, 2.093e-7, 2.014e-7, 1.962e-7, 1.819e-7,
02254
                 1.713e-7, 1.51e-7, 1.34e-7, 1.154e-7, 9.89e-8, 8.88e-8, 7.673e-8,
02256
                 6.599e-8, 5.73e-8, 5.081e-8, 4.567e-8, 4.147e-8, 3.773e-8,
                 3.46e-8, 3.194e-8, 2.953e-8, 2.759e-8, 2.594e-8, 2.442e-8,
02257
02258
                2.355e-8, 2.283e-8, 2.279e-8, 2.231e-8, 2.279e-8, 2.239e-8,
                2.21e-8, 2.309e-8, 2.293e-8, 2.352e-8, 2.415e-8, 2.43e-8, 2.426e-8, 2.465e-8, 2.5e-8, 2.496e-8, 2.465e-8, 2.445e-8, 2.383e-8, 2.299e-8, 2.165e-8, 2.113e-8, 1.968e-8, 1.819e-8,
02259
02260
02261
                 1.644e-8, 1.427e-8, 1.27e-8, 1.082e-8, 9.428e-9, 8.091e-9,
02262
                 6.958e-9, 5.988e-9, 5.246e-9, 4.601e-9, 4.098e-9, 3.664e-9,
02263
02264
                3.287e-9, 2.942e-9, 2.656e-9, 2.364e-9, 2.118e-9, 1.903e-9,
02265
                1.703e-9, 1.525e-9, 1.365e-9, 1.229e-9, 1.107e-9, 9.96e-10,
                8.945e-10, 8.08e-10, 7.308e-10, 6.616e-10, 5.994e-10, 5.422e-10, 4.929e-10, 4.478e-10, 4.07e-10, 3.707e-10, 3.379e-10, 3.087e-10,
02266
02267
02268
                2.823e-10, 2.592e-10, 2.385e-10, 2.201e-10, 2.038e-10, 1.897e-10,
                 1.774e-10, 1.667e-10, 1.577e-10, 1.502e-10, 1.437e-10, 1.394e-10,
02269
02270
                1.358e-10, 1.324e-10, 1.329e-10, 1.324e-10, 1.36e-10, 1.39e-10,
02271
                 1.424e-10, 1.544e-10, 1.651e-10, 1.817e-10, 1.984e-10, 2.195e-10,
                2.438e-10, 2.7e-10, 2.991e-10, 3.322e-10, 3.632e-10, 3.957e-10, 4.36e-10, 4.701e-10, 5.03e-10, 5.381e-10, 5.793e-10, 6.19e-10,
02272
02273
                 6.596e-10, 7.004e-10, 7.561e-10, 7.934e-10, 8.552e-10, 9.142e-10,
02275
                 9.57e-10, 1.027e-9, 1.097e-9, 1.193e-9, 1.334e-9, 1.47e-9,
                 1.636e-9, 1.871e-9, 2.122e-9, 2.519e-9, 2.806e-9, 3.203e-9,
02276
02277
                3.846e-9, 4.362e-9, 5.114e-9, 5.643e-9, 6.305e-9, 6.981e-9,
                7.983e-9, 8.783e-9, 9.419e-9, 1.017e-8, 1.063e-8, 1.121e-8, 1.13e-8, 1.201e-8, 1.225e-8, 1.232e-8, 1.223e-8, 1.177e-8, 1.151e-8, 1.116e-8, 1.047e-8, 9.698e-9, 8.734e-9, 8.202e-9,
02278
02279
02280
                7.041e-9, 6.074e-9, 5.172e-9, 4.468e-9, 3.913e-9, 3.414e-9, 2.975e-9, 2.65e-9, 2.406e-9, 2.173e-9, 2.009e-9, 1.861e-9,
02281
02282
02283
                1.727e-9, 1.612e-9, 1.514e-9, 1.43e-9, 1.362e-9, 1.333e-9,
02284
                1.288e-9, 1.249e-9, 1.238e-9, 1.228e-9, 1.217e-9, 1.202e-9,
                 1.209e-9, 1.177e-9, 1.157e-9, 1.165e-9, 1.142e-9, 1.131e-9,
02285
02286
                 1.138e-9, 1.117e-9, 1.1e-9, 1.069e-9, 1.023e-9, 1.005e-9,
                 9.159e-10, 8.863e-10, 7.865e-10, 7.153e-10, 6.247e-10, 5.846e-10,
                 5.133e-10, 4.36e-10, 3.789e-10, 3.335e-10, 2.833e-10, 2.483e-10,
02288
02289
                2.155e-10, 1.918e-10, 1.709e-10, 1.529e-10, 1.374e-10, 1.235e-10,
02290
                1.108e-10,\ 9.933e-11,\ 8.932e-11,\ 8.022e-11,\ 7.224e-11,\ 6.52e-11,
02291
                5.896e-11, 5.328e-11, 4.813e-11, 4.365e-11, 3.961e-11, 3.594e-11,
                3.266e-11, 2.967e-11, 2.701e-11, 2.464e-11, 2.248e-11, 2.054e-11,
02292
                 1.878e-11, 1.721e-11, 1.579e-11, 1.453e-11, 1.341e-11, 1.241e-11,
                 1.154e-11, 1.078e-11, 1.014e-11, 9.601e-12, 9.167e-12, 8.838e-12,
02294
02295
                8.614e-12, 8.493e-12, 8.481e-12, 8.581e-12, 8.795e-12, 9.131e-12,
02296
                9.601e-12, 1.021e-11, 1.097e-11, 1.191e-11, 1.303e-11, 1.439e-11,
02297
                1.601e-11, 1.778e-11, 1.984e-11, 2.234e-11, 2.474e-11, 2.766e-11,
                3.085e-11, 3.415e-11, 3.821e-11, 4.261e-11, 4.748e-11, 5.323e-11, 5.935e-11, 6.619e-11, 7.418e-11, 8.294e-11, 9.26e-11, 1.039e-10,
02298
                 1.156e-10, 1.297e-10, 1.46e-10, 1.641e-10, 1.858e-10, 2.1e-10,
02300
02301
                2.383e-10, 2.724e-10, 3.116e-10, 3.538e-10, 4.173e-10, 4.727e-10,
                5.503e-10, 6.337e-10, 7.32e-10, 8.298e-10, 9.328e-10, 1.059e-9, 1.176e-9, 1.328e-9, 1.445e-9, 1.593e-9, 1.77e-9, 1.954e-9, 2.175e-9, 2.405e-9, 2.622e-9, 2.906e-9, 3.294e-9, 3.713e-9,
02302
02303
02304
                7.1326 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406 7, 2.1406
02305
02306
                 9.415e-9, 9.511e-9, 9.704e-9, 9.588e-9, 9.45e-9, 9.086e-9,
02307
02308
                8.798e-9, 8.469e-9, 7.697e-9, 7.168e-9, 6.255e-9, 5.772e-9,
                4.97e-9, 4.27le-9, 3.653e-9, 3.154e-9, 2.742e-9, 2.435e-9, 2.166e-9, 1.936e-9, 1.73le-9, 1.556e-9, 1.399e-9, 1.272e-9, 1.157e-9, 1.066e-9, 9.844e-10, 9.258e-10, 8.787e-10, 8.42le-10,
02309
02310
02311
                 8.083e-10, 8.046e-10, 8.067e-10, 8.181e-10, 8.325e-10, 8.517e-10,
02312
                 9.151e-10, 9.351e-10, 9.677e-10, 1.071e-9, 1.126e-9, 1.219e-9,
02313
02314
                 1.297e-9, 1.408e-9, 1.476e-9, 1.517e-9, 1.6e-9, 1.649e-9,
                1.678e-9, 1.746e-9, 1.742e-9, 1.728e-9, 1.699e-9, 1.655e-9,
02315
                 1.561e-9. 1.48e-9. 1.451e-9. 1.411e-9. 1.171e-9. 1.106e-9.
02316
                9.714e-10, 8.523e-10, 7.346e-10, 6.241e-10, 5.371e-10, 4.704e-10, 4.144e-10, 3.683e-10, 3.292e-10, 2.942e-10, 2.62e-10, 2.341e-10,
02317
                 2.104e-10, 1.884e-10, 1.7e-10, 1.546e-10, 1.394e-10, 1.265e-10,
02319
02320
                 1.14e-10, 1.019e-10, 9.279e-11, 8.283e-11, 7.458e-11, 6.668e-11,
02321
                5.976e-11,\ 5.33e-11,\ 4.794e-11,\ 4.289e-11,\ 3.841e-11,\ 3.467e-11,
                3.13e-11, 2.832e-11, 2.582e-11, 2.356e-11, 2.152e-11, 1.97e-11, 1.808e-11, 1.664e-11, 1.539e-11, 1.434e-11, 1.344e-11, 1.269e-11,
02322
02323
```

```
1.209e-11, 1.162e-11, 1.129e-11, 1.108e-11, 1.099e-11, 1.103e-11,
                  1.119e-11, 1.148e-11, 1.193e-11, 1.252e-11, 1.329e-11, 1.421e-11, 1.555e-11, 1.685e-11, 1.839e-11, 2.054e-11, 2.317e-11, 2.571e-11,
02325
02326
02327
                  2.839e-11,\ 3.171e-11,\ 3.49e-11,\ 3.886e-11,\ 4.287e-11,\ 4.645e-11,
02328
                  5.047e-11, 5.592e-11, 6.109e-11, 6.628e-11, 7.381e-11, 8.088e-11, 8.966e-11, 1.045e-10, 1.12e-10, 1.287e-10, 1.486e-10, 1.662e-10,
02329
                  1.866e-10, 2.133e-10, 2.524e-10, 2.776e-10, 3.204e-10, 3.559e-10,
                  4.028e-10, 4.448e-10, 4.882e-10, 5.244e-10, 5.605e-10, 6.018e-10,
02331
02332
                  6.328e-10, 6.579e-10, 6.541e-10, 7.024e-10, 7.074e-10, 7.068e-10,
02333
                  7.009e-10, 6.698e-10, 6.545e-10, 6.209e-10, 5.834e-10, 5.412e-10,
                  5.001e-10, 4.231e-10, 3.727e-10, 3.211e-10, 2.833e-10, 2.447e-10,
02334
                  2.097e-10, 1.843e-10, 1.639e-10, 1.449e-10, 1.27e-10, 1.161e-10, 1.033e-10, 9.282e-11, 8.407e-11, 7.639e-11, 7.023e-11, 6.474e-11,
02335
02336
                  5.142e-11, 5.76e-11, 5.568e-11, 5.472e-11, 5.39e-11, 5.455e-11, 5.54e-11, 5.58e-11, 6.23e-11, 6.49e-11, 6.868e-11, 7.382e-11,
02337
02338
                 8.022e-11, 8.372e-11, 9.243e-11, 1.004e-10, 1.062e-10, 1.13e-10, 1.176e-10, 1.244e-10, 1.279e-10, 1.298e-10, 1.302e-10, 1.312e-10,
02339
02340
                  1.295e-10, 1.244e-10, 1.211e-10, 1.167e-10, 1.098e-10, 9.927e-11,
02341
                  8.854e-11, 8.011e-11, 7.182e-11, 5.923e-11, 5.212e-11, 4.453e-11,
02343
                  3.832e-11, 3.371e-11, 2.987e-11, 2.651e-11, 2.354e-11, 2.093e-11,
                  1.863e-11, 1.662e-11, 1.486e-11, 1.331e-11, 1.193e-11, 1.071e-11, 9.628e-12, 8.66e-12, 7.801e-12, 7.031e-12, 6.347e-12, 5.733e-12,
02344
02345
                 5.182e-12, 4.695e-12, 4.26e-12, 3.874e-12, 3.533e-12, 3.235e-12, 2.979e-12, 2.76e-12, 2.579e-12, 2.432e-12, 2.321e-12, 2.246e-12, 2.205e-12, 2.196e-12, 2.223e-12, 2.288e-12, 2.387e-12, 2.525e-12,
02346
02347
02348
02349
                  2.704e-12, 2.925e-12, 3.191e-12, 3.508e-12, 3.876e-12, 4.303e-12,
                  4.793e-12, 5.347e-12, 5.978e-12, 6.682e-12, 7.467e-12, 8.34e-12,
02350
02351
                  9.293e-12, 1.035e-11, 1.152e-11, 1.285e-11, 1.428e-11, 1.586e-11,
                 1.764e-11, 1.972e-11, 2.214e-11, 2.478e-11, 2.776e-11, 3.151e-11, 3.591e-11, 4.103e-11, 4.66e-11, 5.395e-11, 6.306e-11, 7.172e-11, 8.358e-11, 9.67e-11, 1.11e-10, 1.325e-10, 1.494e-10, 1.736e-10,
02352
02353
02354
02355
                  2.007e-10, 2.296e-10, 2.608e-10, 3.004e-10, 3.361e-10, 3.727e-10,
                  4.373e-10, 4.838e-10, 5.483e-10, 6.006e-10, 6.535e-10, 6.899e-10,
02356
02357
                  7.687e-10, 8.444e-10, 8.798e-10, 9.135e-10, 9.532e-10, 9.757e-10,
                 9.968e-10, 1.006e-9, 9.949e-10, 9.789e-10, 9.64e-10, 9.215e-10, 8.51e-10, 8.394e-10, 7.707e-10, 7.152e-10, 6.274e-10, 5.598e-10, 5.028e-10, 4.3e-10, 3.71e-10, 3.245e-10, 2.809e-10, 2.461e-10, 2.154e-10, 1.91e-10, 1.685e-10, 1.487e-10, 1.313e-10, 1.163e-10,
02358
02359
02360
02362
                  1.031e-10, 9.172e-11, 8.221e-11, 7.382e-11, 6.693e-11, 6.079e-11,
                  5.581e-11, 5.167e-11, 4.811e-11, 4.506e-11, 4.255e-11, 4.083e-11,
02363
                 3.949e-11, 3.881e-11, 3.861e-11, 3.858e-11, 3.951e-11, 4.045e-11, 4.24e-11, 4.487e-11, 4.806e-11, 5.133e-11, 5.518e-11, 5.919e-11,
02364
02365
                  6.533e-11, 7.031e-11, 7.762e-11, 8.305e-11, 9.252e-11, 9.727e-11, 1.045e-10, 1.117e-10, 1.2e-10, 1.275e-10, 1.341e-10, 1.362e-10,
02366
02367
                  1.438e-10, 1.45e-10, 1.455e-10, 1.455e-10, 1.434e-10, 1.381e-10,
02368
02369
                  1.301e-10, 1.276e-10, 1.163e-10, 1.089e-10, 9.911e-11, 8.943e-11,
02370
                  7.618e-11, 6.424e-11, 5.717e-11, 4.866e-11, 4.257e-11, 3.773e-11,
                 3.331e-11, 2.958e-11, 2.629e-11, 2.316e-11, 2.073e-11, 1.841e-11, 1.635e-11, 1.464e-11, 1.31e-11, 1.16e-11, 1.047e-11, 9.408e-12, 8.414e-12, 7.521e-12, 6.705e-12, 5.993e-12, 5.371e-12, 4.815e-12,
02371
02372
02373
                  4.338e-12, 3.921e-12, 3.567e-12, 3.265e-12, 3.01e-12, 2.795e-12,
                  2.613e-12, 2.464e-12, 2.346e-12, 2.256e-12, 2.195e-12, 2.165e-12,
02375
02376
                  2.166e-12, 2.198e-12, 2.262e-12, 2.364e-12, 2.502e-12, 2.682e-12,
02377
                  2.908e-12, 3.187e-12, 3.533e-12, 3.946e-12, 4.418e-12, 5.013e-12,
                 2.906e-12, 3.16/e-12, 3.35e-12, 3.34e-12, 4.41e-12, 3.35e-12, 5.37e-12, 5.37
02378
02379
                  4.951e-11, 5.452e-11, 6.014e-11, 6.5e-11, 6.915e-11, 7.45e-11,
02381
                  7.971e-11, 8.468e-11, 8.726e-11, 8.995e-11, 9.182e-11, 9.509e-11, 9.333e-11, 9.386e-11, 9.457e-11, 9.21e-11, 9.019e-11, 8.68e-11, 8.298e-11, 7.947e-11, 7.46e-11, 7.082e-11, 6.132e-11, 5.855e-11,
02382
02383
02384
                 5.073e-11, 4.464e-11, 3.825e-11, 3.375e-11, 2.911e-11, 2.535e-11, 2.16e-11, 1.907e-11, 1.665e-11, 1.463e-11, 1.291e-11, 1.133e-11,
02385
02386
                  9.997e-12, 8.836e-12, 7.839e-12, 6.943e-12, 6.254e-12, 5.6e-12,
02387
02388
                  5.029e-12, 4.529e-12, 4.102e-12, 3.737e-12, 3.428e-12, 3.169e-12,
02389
                  2.959e-12, 2.798e-12, 2.675e-12, 2.582e-12, 2.644e-12, 2.557e-12,
02390
                  2.614e-12, 2.717e-12, 2.874e-12, 3.056e-12, 3.187e-12, 3.631e-12, 3.979e-12, 4.248e-12, 4.817e-12, 5.266e-12, 5.836e-12, 6.365e-12,
02391
                  6.807e-12, 7.47e-12, 7.951e-12, 8.636e-12, 8.972e-12, 9.314e-12,
02392
                  9.445e-12, 1.003e-11, 1.013e-11, 9.937e-12, 9.729e-12, 9.064e-12,
                  9.119e-12, 9.124e-12, 8.704e-12, 8.078e-12, 7.47e-12, 6.329e-12,
02394
02395
                  5.674e-12, 4.808e-12, 4.119e-12, 3.554e-12, 3.103e-12, 2.731e-12,
                 2.415e-12, 2.15e-12, 1.926e-12, 1.737e-12, 1.578e-12, 1.447e-12, 1.34e-12, 1.255e-12, 1.191e-12, 1.146e-12, 1.121e-12, 1.114e-12,
02396
02397
02398
                  1.126e-12, 1.156e-12, 1.207e-12, 1.278e-12, 1.372e-12, 1.49e-12,
                  1.633e-12, 1.805e-12, 2.01e-12, 2.249e-12, 2.528e-12, 2.852e-12,
02399
                  3.228e-12, 3.658e-12, 4.153e-12, 4.728e-12, 5.394e-12, 6.176e-12,
02400
02401
                  7.126e-12, 8.188e-12, 9.328e-12, 1.103e-11, 1.276e-11, 1.417e-11,
                  1.615e-11, 1.84e-11, 2.155e-11, 2.429e-11, 2.826e-11, 3.222e-11,
02402
                  3.664e-11, 4.14e-11, 4.906e-11, 5.536e-11, 6.327e-11, 7.088e-11,
02403
                  8.316e-11, 9.242e-11, 1.07e-10, 1.223e-10, 1.341e-10, 1.553e-10, 1.703e-10, 1.9e-10, 2.022e-10, 2.233e-10, 2.345e-10, 2.438e-10,
02404
                  2.546e-10, 2.599e-10, 2.661e-10, 2.703e-10, 2.686e-10, 2.662e-10, 2.56e-10, 2.552e-10, 2.378e-10, 2.252e-10, 2.146e-10, 1.885e-10,
02406
02407
02408
                  1.668e-10, 1.441e-10, 1.295e-10, 1.119e-10, 9.893e-11, 8.687e-11,
02409
                  7.678e-11, 6.685e-11, 5.879e-11, 5.127e-11, 4.505e-11, 3.997e-11,
                  3.511e-11
02410
```

```
02411
02412
          static double h2ofrn[2001] = { .01095, .01126, .01205, .01322, .0143,
02413
            .01506, .01548, .01534, .01486, .01373, .01262, .01134, .01001, .008702, .007475, .006481, .00548, .0046, .003833, .00311, .002543, .002049, .00168, .001374, .001046, 8.193e-4, 6.267e-4,
02414
02415
02416
             4.968e-4, 3.924e-4, 2.983e-4, 2.477e-4, 1.997e-4, 1.596e-4,
02417
             1.331e-4, 1.061e-4, 8.942e-5, 7.168e-5, 5.887e-5, 4.848e-5
02418
02/19
            3.817e-5, 3.17e-5, 2.579e-5, 2.162e-5, 1.768e-5, 1.49e-5,
            1.231e-5, 1.013e-5, 8.555e-6, 7.328e-6, 6.148e-6, 5.207e-6, 4.387e-6, 3.741e-6, 3.22e-6, 2.753e-6, 2.346e-6, 1.985e-6,
02420
02421
02422
             1.716e-6, 1.475e-6, 1.286e-6, 1.122e-6, 9.661e-7, 8.284e-7,
            7.057e-7, 6.119e-7, 5.29e-7, 4.571e-7, 3.948e-7, 3.432e-7, 2.983e-7, 2.589e-7, 2.265e-7, 1.976e-7, 1.704e-7, 1.456e-7,
02423
02424
             1.26e-7, 1.101e-7, 9.648e-8, 8.415e-8, 7.34e-8, 6.441e-8,
02425
02426
             5.643e-8, 4.94e-8, 4.276e-8, 3.703e-8, 3.227e-8, 2.825e-8,
            2.478e-8, 2.174e-8, 1.898e-8, 1.664e-8, 1.458e-8, 1.278e-8, 1.126e-8, 9.891e-9, 8.709e-9, 7.652e-9, 6.759e-9, 5.975e-9,
02427
02428
             5.31e-9, 4.728e-9, 4.214e-9, 3.792e-9, 3.463e-9, 3.226e-9,
             2.992e-9, 2.813e-9, 2.749e-9, 2.809e-9, 2.913e-9, 3.037e-9,
02430
             3.413e-9, 3.738e-9, 4.189e-9, 4.808e-9, 5.978e-9, 7.088e-9,
02431
02432
            8.071e-9, 9.61e-9, 1.21e-8, 1.5e-8, 1.764e-8, 2.221e-8, 2.898e-8,
            3.948e-8, 5.068e-8, 6.227e-8, 7.898e-8, 1.033e-7, 1.437e-7, 1.889e-7, 2.589e-7, 3.59e-7, 4.971e-7, 7.156e-7, 9.983e-7, 1.381e-6, 1.929e-6, 2.591e-6, 3.453e-6, 4.57e-6, 5.93e-6,
02433
02434
02435
02436
             7.552e-6, 9.556e-6, 1.183e-5, 1.425e-5, 1.681e-5, 1.978e-5
02437
            2.335e-5, 2.668e-5, 3.022e-5, 3.371e-5, 3.715e-5, 3.967e-5,
02438
             4.06e-5, 4.01e-5, 3.809e-5, 3.491e-5, 3.155e-5, 2.848e-5,
            2.678e-5, 2.66e-5, 2.811e-5, 3.071e-5, 3.294e-5, 3.459e-5, 3.569e-5, 3.56e-5, 3.434e-5, 3.186e-5, 2.916e-5, 2.622e-5,
02439
02440
            2.275e-5, 1.918e-5, 1.62e-5, 1.373e-5, 1.182e-5, 1.006e-5, 8.556e-6, 7.26e-6, 6.107e-6, 5.034e-6, 4.211e-6, 3.426e-6,
02441
02442
02443
             2.865e-6, 2.446e-6, 1.998e-6, 1.628e-6, 1.242e-6, 1.005e-6,
02444
             7.853e-7, 6.21e-7, 5.071e-7, 4.156e-7, 3.548e-7, 2.825e-7,
            2.261e-7, 1.916e-7, 1.51e-7, 1.279e-7, 1.059e-7, 9.14e-8, 7.707e-8, 6.17e-8, 5.311e-8, 4.263e-8, 3.518e-8, 2.961e-8,
02445
02446
             2.457e-8, 2.119e-8, 1.712e-8, 1.439e-8, 1.201e-8, 1.003e-8,
02447
             8.564e-9, 7.199e-9, 6.184e-9, 5.206e-9, 4.376e-9, 3.708e-9,
02449
             3.157e-9, 2.725e-9, 2.361e-9, 2.074e-9, 1.797e-9,
                                                                             1.562e-9.
02450
             1.364e-9, 1.196e-9, 1.042e-9, 8.862e-10, 7.648e-10, 6.544e-10,
02451
            5.609e-10, 4.791e-10, 4.108e-10, 3.531e-10, 3.038e-10, 2.618e-10,
            2.268e-10, 1.969e-10, 1.715e-10, 1.496e-10, 1.308e-10, 1.147e-10,
02452
            1.008e-10, 8.894e-11, 7.885e-11, 7.031e-11, 6.355e-11, 5.854e-11, 5.534e-11, 5.466e-11, 5.725e-11, 6.447e-11, 7.943e-11, 1.038e-10,
02453
02454
             1.437e-10, 2.04e-10, 2.901e-10, 4.051e-10, 5.556e-10, 7.314e-10,
02455
02456
             9.291e-10, 1.134e-9, 1.321e-9, 1.482e-9, 1.596e-9, 1.669e-9,
02457
            1.715e-9, 1.762e-9, 1.817e-9, 1.828e-9, 1.848e-9, 1.873e-9,
            1.902e-9, 1.894e-9, 1.864e-9, 1.841e-9, 1.797e-9, 1.704e-9,
02458
             1.559e-9, 1.382e-9, 1.187e-9, 1.001e-9, 8.468e-10, 7.265e-10,
02459
02460
             6.521e-10, 6.381e-10, 6.66e-10, 7.637e-10, 9.705e-10, 1.368e-9,
             1.856e-9, 2.656e-9, 3.954e-9, 5.96e-9, 8.72e-9, 1.247e-8,
             1.781e-8, 2.491e-8, 3.311e-8, 4.272e-8, 5.205e-8, 6.268e-8,
02462
02463
             7.337e-8, 8.277e-8, 9.185e-8, 1.004e-7, 1.091e-7, 1.159e-7,
02464
             1.188e-7, 1.175e-7, 1.124e-7, 1.033e-7, 9.381e-8, 8.501e-8,
            7.956-8, 7.894e-8, 8.331e-8, 9.102e-8, 9.836e-8, 1.035e-7, 1.064e-7, 1.06e-7, 1.032e-7, 9.808e-8, 9.139e-8, 8.442e-8,
02465
02466
             7.641e-8, 6.881e-8, 6.161e-8, 5.404e-8, 4.804e-8, 4.446e-8,
02467
             4.328e-8, 4.259e-8, 4.421e-8, 4.673e-8, 4.985e-8, 5.335e-8,
02468
02469
             5.796e-8, 6.542e-8, 7.714e-8, 8.827e-8, 1.04e-7, 1.238e-7,
            1.499e-7, 1.829e-7, 2.222e-7, 2.689e-7, 3.303e-7, 3.981e-7, 4.84e-7, 5.91e-7, 7.363e-7, 9.087e-7, 1.139e-6, 1.455e-6,
02470
02471
            1.866e-6, 2.44e-6, 3.115e-6, 3.941e-6, 4.891e-6, 5.992e-6, 7.111e-6, 8.296e-6, 9.21e-6, 9.987e-6, 1.044e-5, 1.073e-5,
02472
             1.092e-5, 1.106e-5, 1.138e-5, 1.171e-5, 1.186e-5, 1.186e-5,
02474
02475
            1.179e-5, 1.166e-5, 1.151e-5, 1.16e-5, 1.197e-5, 1.241e-5,
02476
            1.268e-5, 1.26e-5, 1.184e-5, 1.063e-5, 9.204e-6, 7.584e-6,
            6.053e-6, 4.482e-6, 3.252e-6, 2.337e-6, 1.662e-6, 1.18e-6, 8.15e-7, 5.95e-7, 4.354e-7, 3.302e-7, 2.494e-7, 1.93e-7, 1.545e-7, 1.25e-7, 1.039e-7, 8.602e-8, 7.127e-8, 5.897e-8,
02477
02478
02479
             4.838e-8, 4.018e-8, 3.28e-8, 2.72e-8, 2.307e-8, 1.972e-8,
             1.654e-8, 1.421e-8, 1.174e-8, 1.004e-8, 8.739e-9, 7.358e-9,
02481
02482
             6.242e-9, 5.303e-9, 4.567e-9, 3.94e-9, 3.375e-9, 2.864e-9,
            2.422e-9, 2.057e-9, 1.75e-9, 1.505e-9, 1.294e-9, 1.101e-9, 9.401e-10, 8.018e-10, 6.903e-10, 5.965e-10, 5.087e-10, 4.364e-10, 3.759e-10, 3.247e-10, 2.809e-10, 2.438e-10, 2.123e-10, 1.853e-10,
02483
02484
02485
             1.622e-10, 1.426e-10, 1.26e-10, 1.125e-10, 1.022e-10, 9.582e-11,
02486
             9.388e-11, 9.801e-11, 1.08e-10, 1.276e-10, 1.551e-10, 1.903e-10,
02487
02488
             2.291e-10, 2.724e-10, 3.117e-10, 3.4e-10, 3.562e-10, 3.625e-10,
02489
            3.619e-10, 3.429e-10, 3.221e-10, 2.943e-10, 2.645e-10, 2.338e-10,
            2.062e-10, 1.901e-10, 1.814e-10, 1.827e-10, 1.906e-10, 1.984e-10,
02490
            2.04e-10, 2.068e-10, 2.075e-10, 2.018e-10, 1.959e-10, 1.897e-10, 1.852e-10, 1.791e-10, 1.696e-10, 1.634e-10, 1.598e-10, 1.561e-10,
02491
             1.518e-10, 1.443e-10, 1.377e-10, 1.346e-10, 1.342e-10, 1.375e-10,
02493
02494
            1.525e-10, 1.767e-10, 2.108e-10, 2.524e-10, 2.981e-10, 3.477e-10,
02495
            4.262e-10, 5.326e-10, 6.646e-10, 8.321e-10, 1.069e-9, 1.386e-9,
            1.743e-9, 2.216e-9, 2.808e-9, 3.585e-9, 4.552e-9, 5.907e-9, 7.611e-9, 9.774e-9, 1.255e-8, 1.666e-8, 2.279e-8, 3.221e-8,
02496
02497
```

```
4.531e-8, 6.4e-8, 9.187e-8, 1.295e-7, 1.825e-7, 2.431e-7,
              3.181e-7, 4.009e-7, 4.941e-7, 5.88e-7, 6.623e-7, 7.155e-7, 7.451e-7, 7.594e-7, 7.541e-7, 7.467e-7, 7.527e-7, 7.935e-7,
02499
02500
              8.461e-7, 8.954e-7, 9.364e-7, 9.843e-7, 1.024e-6, 1.05e-6,
02501
              1.059e-6, 1.074e-6, 1.072e-6, 1.043e-6, 9.789e-7, 8.803e-7, 7.662e-7, 6.378e-7, 5.133e-7, 3.958e-7, 2.914e-7, 2.144e-7
02502
02503
              1.57e-7, 1.14e-7, 8.47e-8, 6.2e-8, 4.657e-8, 3.559e-8, 2.813e-8,
              2.222e-8, 1.769e-8, 1.391e-8, 1.125e-8, 9.186e-9, 7.704e-9,
02505
02506
              6.447e-9, 5.381e-9, 4.442e-9, 3.669e-9, 3.057e-9, 2.564e-9
             8.169e-10, 6.856e-10, 5.866e-10, 5.043e-10, 4.336e-10, 3.731e-10, 3.175e-10, 2.745e-10, 2.374e-10, 2.007e-10, 1.737e-10, 1.508e-10, 1.302e-10, 1.13e-10, 9.672e-11, 8.375e-11, 7.265e-11, 6.244e-11,
02507
02508
02509
02510
02511
              5.343e-11, 4.654e-11, 3.975e-11, 3.488e-11, 3.097e-11, 2.834e-11,
02512
              2.649e-11, 2.519e-11, 2.462e-11, 2.443e-11, 2.44e-11, 2.398e-11,
              2.306e-11, 2.183e-11, 2.021e-11, 1.821e-11, 1.599e-11, 1.403e-11, 1.196e-11, 1.023e-11, 8.728e-12, 7.606e-12, 6.941e-12, 6.545e-12,
02513
02514
              6.484e-12, 6.6e-12, 6.718e-12, 6.785e-12, 6.746e-12, 6.724e-12,
02515
              6.764e-12, 6.995e-12, 7.144e-12, 7.32e-12, 7.33e-12, 7.208e-12,
              6.789e-12, 6.09e-12, 5.337e-12, 4.62e-12, 4.037e-12, 3.574e-12,
02517
02518
              3.311e-12, 3.346e-12, 3.566e-12, 3.836e-12, 4.076e-12, 4.351e-12,
02519
              4.691e-12, 5.114e-12, 5.427e-12, 6.167e-12, 7.436e-12, 8.842e-12,
              1.038e-11, 1.249e-11, 1.54e-11, 1.915e-11, 2.48e-11, 3.256e-11,
02520
              4.339e-11, 5.611e-11, 7.519e-11, 1.037e-10, 1.409e-10, 1.883e-10, 2.503e-10, 3.38e-10, 4.468e-10, 5.801e-10, 7.335e-10, 8.98e-10,
02521
              1.11e-9, 1.363e-9, 1.677e-9, 2.104e-9, 2.681e-9, 3.531e-9,
              4.621e-9, 6.106e-9, 8.154e-9, 1.046e-8, 1.312e-8, 1.607e-8,
02524
02525
              1.948e-8, 2.266e-8, 2.495e-8, 2.655e-8, 2.739e-8, 2.739e-8,
             2.662e-8, 2.589e-8, 2.59e-8, 2.664e-8, 2.833e-8, 3.023e-8, 3.305e-8, 3.558e-8, 3.793e-8, 3.961e-8, 4.056e-8, 4.102e-8,
02526
02527
02528
              4.025e-8, 3.917e-8, 3.706e-8, 3.493e-8, 3.249e-8, 3.096e-8,
              3.011e-8, 3.111e-8, 3.395e-8, 3.958e-8, 4.875e-8, 6.066e-8,
             7.915e-8, 1.011e-7, 1.3e-7, 1.622e-7, 2.03e-7, 2.448e-7, 2.863e-7, 3.317e-7, 3.655e-7, 3.96e-7, 4.098e-7, 4.168e-7,
02530
02531
              4.198e-7, 4.207e-7, 4.289e-7, 4.384e-7, 4.471e-7, 4.524e-7, 4.574e-7, 4.633e-7, 4.785e-7, 5.028e-7, 5.371e-7, 5.727e-7,
02532
02533
              5.955e-7, 5.998e-7, 5.669e-7, 5.082e-7, 4.397e-7, 3.596e-7, 2.814e-7, 2.074e-7, 1.486e-7, 1.057e-7, 7.25e-8, 4.946e-8,
02534
02536
              3.43e-8, 2.447e-8, 1.793e-8, 1.375e-8, 1.096e-8, 9.091e-9,
              7.709e-9, 6.631e-9, 5.714e-9, 4.886e-9, 4.205e-9, 3.575e-9,
02537
              3.07e-9, 2.631e-9, 2.284e-9, 2.002e-9, 1.745e-9, 1.509e-9,
02538
              1.284e-9, 1.084e-9, 9.163e-10, 7.663e-10, 6.346e-10, 5.283e-10,
02539
              1.354e-10, 3.59e-10, 2.982e-10, 2.455e-10, 2.033e-11, 1.696e-10, 1.432e-10, 1.211e-10, 1.02e-10, 8.702e-11, 7.38e-11, 6.293e-11,
02540
              5.343e-11, 4.532e-11, 3.907e-11, 3.365e-11, 2.945e-11, 2.558e-11, 2.192e-11, 1.895e-11, 1.636e-11, 1.42e-11, 1.228e-11, 1.063e-11,
02542
02543
             2.132e 11, 1.035e 11, 1.036e 11, 1.132e 11, 1.22e 11, 1.035e 12, 3.137e-12, 3.197e-12, 5.052e-12, 4.469e-12, 4.e-12, 3.679e-12, 3.387e-12, 3.197e-12, 3.158e-12, 3.327e-12, 3.675e-12, 4.292e-12, 5.437e-12, 7.197e-12, 1.008e-11, 1.437e-11, 2.035e-11, 2.905e-11, 4.062e-11, 5.528e-11, 7.177e-11,
02544
02545
02546
02547
              9.064e-11, 1.109e-10, 1.297e-10, 1.473e-10, 1.652e-10, 1.851e-10,
              2.079e-10, 2.313e-10, 2.619e-10, 2.958e-10, 3.352e-10, 3.796e-10,
02549
02550
              4.295e-10, 4.923e-10, 5.49e-10, 5.998e-10, 6.388e-10, 6.645e-10,
              6.712e-10, 6.549e-10, 6.38e-10, 6.255e-10, 6.253e-10, 6.459e-10, 6.977e-10, 7.59e-10, 8.242e-10, 8.92e-10, 9.403e-10, 9.701e-10,
02551
02552
              9.483e-10, 9.135e-10, 8.617e-10, 7.921e-10, 7.168e-10, 6.382e-10,
02553
              5.677e-10, 5.045e-10, 4.572e-10, 4.312e-10, 4.145e-10, 4.192e-10,
              4.541e-10, 5.368e-10, 6.771e-10, 8.962e-10, 1.21e-9, 1.659e-9,
02555
             4.341e-10, 3.366e-10, 6.71e-10, 6.362e-10, 1.21e-3, 1.63e
2.33e-9, 3.249e-9, 4.495e-9, 5.923e-9, 7.642e-9, 9.607e-9,
1.178e-8, 1.399e-8, 1.584e-8, 1.73e-8, 1.816e-8, 1.87e-8,
1.868e-8, 1.87e-8, 1.884e-8, 1.99e-8, 2.15e-8, 2.258e-8,
2.364e-8, 2.473e-8, 2.602e-8, 2.689e-8, 2.731e-8, 2.816e-8,
2.859e-8, 2.839e-8, 2.703e-8, 2.451e-8, 2.149e-8, 1.787e-8,
02556
02557
02558
02559
              1.449e-8, 1.111e-8, 8.282e-9, 6.121e-9, 4.494e-9, 3.367e-9,
02561
02562
              2.487e-9, 1.885e-9, 1.503e-9, 1.249e-9, 1.074e-9, 9.427e-10,
02563
              8.439e-10, 7.563e-10, 6.772e-10, 6.002e-10, 5.254e-10, 4.588e-10,
02564
             3.977e-10, 3.449e-10, 3.003e-10, 2.624e-10, 2.335e-10, 2.04e-10, 1.771e-10, 1.534e-10, 1.296e-10, 1.097e-10, 9.173e-11, 7.73e-11,
02565
02566
              6.547e-11, 5.191e-11, 4.198e-11, 3.361e-11, 2.732e-11, 2.244e-11,
              1.791e-11, 1.509e-11, 1.243e-11, 1.035e-11, 8.969e-12, 7.394e-12,
              6.323e-12, 5.282e-12, 4.543e-12, 3.752e-12, 3.14e-12, 2.6e-12,
02568
02569
              2.194e-12, 1.825e-12, 1.511e-12, 1.245e-12, 1.024e-12, 8.539e-13,
              7.227e-13, 6.102e-13, 5.189e-13, 4.43e-13, 3.774e-13, 3.236e-13, 2.8e-13, 2.444e-13, 2.156e-13, 1.932e-13, 1.775e-13, 1.695e-13,
02570
02571
              1.672e-13, 1.704e-13, 1.825e-13, 2.087e-13, 2.614e-13, 3.377e-13, 4.817e-13, 6.989e-13, 1.062e-12, 1.562e-12, 2.288e-12, 3.295e-12,
02572
02573
              4.55e-12, 5.965e-12, 7.546e-12, 9.395e-12, 1.103e-11, 1.228e-11,
02574
02575
              1.318e-11, 1.38e-11, 1.421e-11, 1.39e-11, 1.358e-11, 1.336e-11,
02576
              1.342e-11, 1.356e-11, 1.424e-11, 1.552e-11, 1.73e-11, 1.951e-11,
              2.128e-11, 2.249e-11, 2.277e-11, 2.226e-11, 2.111e-11, 1.922e-11,
02577
02578
              1.775e-11, 1.661e-11, 1.547e-11, 1.446e-11, 1.323e-11, 1.21e-11,
              1.054e-11, 9.283e-12, 8.671e-12, 8.67e-12, 9.429e-12, 1.062e-11,
              1.255e-11, 1.506e-11, 1.818e-11, 2.26e-11, 2.831e-11, 3.723e-11, 5.092e-11, 6.968e-11, 9.826e-11, 1.349e-10, 1.87e-10, 2.58e-10,
02580
02581
02582
              3.43 e^{-10},\ 4.424 e^{-10},\ 5.521 e^{-10},\ 6.812 e^{-10},\ 8.064 e^{-10},\ 9.109 e^{-10},
              9.839e-10, 1.028e-9, 1.044e-9, 1.029e-9, 1.005e-9, 1.002e-9, 1.038e-9, 1.122e-9, 1.233e-9, 1.372e-9, 1.524e-9, 1.665e-9,
02583
02584
```

```
1.804e-9, 1.908e-9, 2.015e-9, 2.117e-9, 2.219e-9, 2.336e-9,
             2.531e-9, 2.805e-9, 3.189e-9, 3.617e-9, 4.208e-9, 4.911e-9, 5.619e-9, 6.469e-9, 7.188e-9, 7.957e-9, 8.503e-9, 9.028e-9,
02586
02587
02588
             9.571e-9, 9.99e-9, 1.055e-8, 1.102e-8, 1.132e-8, 1.141e-8,
             1.145e-8, 1.145e-8, 1.176e-8, 1.224e-8, 1.304e-8, 1.388e-8, 1.445e-8, 1.453e-8, 1.368e-8, 1.22e-8, 1.042e-8, 8.404e-9,
02589
02590
              6.403e-9, 4.643e-9, 3.325e-9, 2.335e-9, 1.638e-9, 1.19e-9,
             9.161e-10, 7.412e-10, 6.226e-10, 5.516e-10, 5.068e-10, 4.831e-10,
02592
02593
             4.856e-10, 5.162e-10, 5.785e-10, 6.539e-10, 7.485e-10, 8.565e-10,
02594
             9.534e-10, 1.052e-9, 1.115e-9, 1.173e-9, 1.203e-9, 1.224e-9,
             1.243e-9, 1.248e-9, 1.261e-9, 1.265e-9, 1.25e-9, 1.217e-9, 1.176e-9, 1.145e-9, 1.153e-9, 1.199e-9, 1.278e-9, 1.366e-9, 1.426e-9, 1.444e-9, 1.365e-9, 1.224e-9, 1.051e-9, 8.539e-10, 6.564e-10, 4.751e-10, 3.404e-10, 2.377e-10, 1.631e-10, 1.114e-10,
02595
02596
02598
02599
             7.87e-11, 5.793e-11, 4.284e-11, 3.3e-11, 2.62e-11, 2.152e-11,
             1.777e-11, 1.496e-11, 1.242e-11, 1.037e-11, 8.725e-12, 7.004e-12, 5.718e-12, 4.769e-12, 3.952e-12, 3.336e-12, 2.712e-12, 2.213e-12,
02600
02601
02602
             1.803e-12, 1.492e-12, 1.236e-12, 1.006e-12, 8.384e-13, 7.063e-13,
             5.879e-13, 4.93e-13, 4.171e-13, 3.569e-13, 3.083e-13, 2.688e-13,
02603
             2.333e-13, 2.035e-13, 1.82e-13, 1.682e-13, 1.635e-13, 1.674e-13,
02604
             1.769e-13, 2.022e-13, 2.485e-13, 3.127e-13, 4.25e-13, 5.928e-13,
02605
02606
             8.514e-13, 1.236e-12, 1.701e-12, 2.392e-12, 3.231e-12, 4.35e-12,
02607
             5.559e-12, 6.915e-12, 8.519e-12, 1.013e-11, 1.146e-11, 1.24e-11,
             1.305e-11, 1.333e-11, 1.318e-11, 1.263e-11, 1.238e-11, 1.244e-11, 1.305e-11, 1.432e-11, 1.623e-11, 1.846e-11, 2.09e-11, 2.328e-11,
02608
02609
             2.526e-11, 2.637e-11, 2.702e-11, 2.794e-11, 2.889e-11, 2.989e-11, 3.231e-11, 3.68e-11, 4.375e-11, 5.504e-11, 7.159e-11, 9.502e-11,
02611
02612
             1.279e-10, 1.645e-10, 2.098e-10, 2.618e-10, 3.189e-10, 3.79e-10,
02613
             4.303e-10, 4.753e-10, 5.027e-10, 5.221e-10, 5.293e-10, 5.346e-10,
             5.467e-10, 5.796e-10, 6.2e-10, 6.454e-10, 6.705e-10, 6.925e-10, 7.233e-10, 7.35e-10, 7.538e-10, 7.861e-10, 8.077e-10, 8.132e-10,
02614
02615
02616
              7.749e-10, 7.036e-10, 6.143e-10, 5.093e-10, 4.089e-10, 3.092e-10,
             2.299e-10, 1.705e-10, 1.277e-10, 9.723e-11, 7.533e-11, 6.126e-11,
02617
02618
             5.154e-11, 4.428e-11, 3.913e-11, 3.521e-11, 3.297e-11, 3.275e-11,
             3.46e-11, 3.798e-11, 4.251e-11, 4.745e-11, 5.232e-11, 5.606e-11, 5.82e-11, 5.88e-11, 5.79e-11, 5.661e-11, 5.491e-11, 5.366e-11, 5.341e-11, 5.353e-11, 5.336e-11, 5.293e-11, 5.248e-11, 5.235e-11,
02619
02620
02621
             5.208e-11, 5.322e-11, 5.521e-11, 5.725e-11, 5.827e-11, 5.685e-11,
02623
             5.245e-11, 4.612e-11, 3.884e-11, 3.129e-11, 2.404e-11, 1.732e-11,
             1.223e-11, 8.574e-12, 5.888e-12, 3.986e-12, 2.732e-12, 1.948e-12,
02624
02625
             1.414e-12, 1.061e-12, 8.298e-13, 6.612e-13, 5.413e-13, 4.472e-13,
             3.772e-13, 3.181e-13, 2.645e-13, 2.171e-13, 1.778e-13, 1.464e-13,
02626
             1.183e-13, 9.637e-14, 7.991e-14, 6.668e-14, 5.57e-14, 4.663e-14, 3.848e-14, 3.233e-14, 2.706e-14, 2.284e-14, 1.944e-14, 1.664e-14,
02627
02628
             1.43e-14, 1.233e-14, 1.066e-14, 9.234e-15, 8.023e-15, 6.993e-15,
02629
02630
             6.119e-15, 5.384e-15, 4.774e-15, 4.283e-15, 3.916e-15, 3.695e-15,
02631
             3.682e-15, 4.004e-15, 4.912e-15, 6.853e-15, 1.056e-14, 1.712e-14,
             2.804e-14, 4.516e-14, 7.113e-14, 1.084e-13, 1.426e-13, 1.734e-13, 1.978e-13, 2.194e-13, 2.388e-13, 2.489e-13, 2.626e-13, 2.865e-13,
02632
02633
02634
             3.105e-13, 3.387e-13, 3.652e-13, 3.984e-13, 4.398e-13, 4.906e-13, 5.55e-13, 6.517e-13, 7.813e-13, 9.272e-13, 1.164e-12, 1.434e-12,
             1.849e-12, 2.524e-12, 3.328e-12, 4.523e-12, 6.108e-12, 8.207e-12,
02636
02637
             1.122e-11, 1.477e-11, 1.9e-11, 2.412e-11, 2.984e-11, 3.68e-11,
             1.353e-11, 4.963e-11, 5.478e-11, 5.903e-11, 6.233e-11, 6.483e-11, 6.904e-11, 7.569e-11, 8.719e-11, 1.048e-10, 1.278e-10, 1.557e-10, 1.869e-10, 2.218e-10, 2.61e-10, 2.975e-10, 3.371e-10, 3.746e-10, 4.065e-10, 4.336e-10, 4.503e-10, 4.701e-10, 4.8e-10, 4.917e-10,
02638
02639
02640
             5.038e-10, 5.128e-10, 5.143e-10, 5.071e-10, 5.019e-10, 5.025e-10,
02642
02643
             5.183e-10, 5.496e-10, 5.877e-10, 6.235e-10, 6.42e-10, 6.234e-10,
02644
             5.698e-10, 4.916e-10, 4.022e-10, 3.126e-10, 2.282e-10, 1.639e-10,
             1.142e-10, 7.919e-11, 5.69e-11, 4.313e-11, 3.413e-11, 2.807e-11, 2.41e-11, 2.166e-11, 2.024e-11, 1.946e-11, 1.929e-11, 1.963e-11,
02645
02646
02647
             2.035e-11, 2.162e-11, 2.305e-11, 2.493e-11, 2.748e-11, 3.048e-11,
             3.413e-11, 3.754e-11, 4.155e-11, 4.635e-11, 5.11e-11, 5.734e-11, 6.338e-11, 6.99e-11, 7.611e-11, 8.125e-11, 8.654e-11, 8.951e-11,
02648
02649
02650
             9.182e-11, 9.31e-11, 9.273e-11, 9.094e-11, 8.849e-11, 8.662e-11,
02651
             8.67e-11, 8.972e-11, 9.566e-11, 1.025e-10, 1.083e-10, 1.111e-10, 1.074e-10, 9.771e-11, 8.468e-11, 6.958e-11, 5.47e-11, 4.04e-11,
02652
             2.94e-11, 2.075e-11, 1.442e-11, 1.01e-11, 7.281e-12, 5.409e-12,
02653
              4.138e-12, 3.304e-12, 2.784e-12, 2.473e-12, 2.273e-12, 2.186e-12,
02655
             2.118e-12, 2.066e-12, 1.958e-12, 1.818e-12, 1.675e-12, 1.509e-12,
02656
             1.349e-12, 1.171e-12, 9.838e-13, 8.213e-13, 6.765e-13, 5.378e-13,
             4.161e-13, 3.119e-13, 2.279e-13, 1.637e-13, 1.152e-13, 8.112e-14, 5.919e-14, 4.47e-14, 3.492e-14, 2.811e-14, 2.319e-14, 1.948e-14, 1.66e-14, 1.432e-14, 1.251e-14, 1.109e-14, 1.006e-14, 9.45e-15,
02657
02658
02659
             9.384e-15, 1.012e-14, 1.216e-14, 1.636e-14, 2.305e-14, 3.488e-14,
02660
             5.572e-14, 8.479e-14, 1.265e-13, 1.905e-13, 2.73e-13, 3.809e-13,
02661
02662
             4.955e-13, 6.303e-13, 7.861e-13, 9.427e-13, 1.097e-12, 1.212e-12,
             1.328e-12, 1.415e-12, 1.463e-12, 1.495e-12, 1.571e-12, 1.731e-12, 1.981e-12, 2.387e-12, 2.93e-12, 3.642e-12, 4.584e-12, 5.822e-12,
02663
02664
             7.278e-12, 9.193e-12, 1.135e-11, 1.382e-11, 1.662e-11, 1.958e-11, 2.286e-11, 2.559e-11, 2.805e-11, 2.988e-11, 3.106e-11, 3.182e-11,
02665
              3.2e-11, 3.258e-11, 3.362e-11, 3.558e-11, 3.688e-11, 3.8e-11,
02667
02668
             3.929e-11, 4.062e-11, 4.186e-11, 4.293e-11, 4.48e-11, 4.643e-11,
02669
             4.704e-11, 4.571e-11, 4.206e-11, 3.715e-11, 3.131e-11, 2.541e-11,
             1.978e-11, 1.508e-11, 1.146e-11, 8.7e-12, 6.603e-12, 5.162e-12, 4.157e-12, 3.408e-12, 2.829e-12, 2.405e-12, 2.071e-12, 1.826e-12,
02670
02671
```

```
1.648e-12, 1.542e-12, 1.489e-12, 1.485e-12, 1.493e-12, 1.545e-12,
           1.637e-12, 1.814e-12, 2.061e-12, 2.312e-12, 2.651e-12, 3.03e-12,
           3.46e-12, 3.901e-12, 4.306e-12, 4.721e-12, 5.008e-12, 5.281e-12,
02674
02675
           5.541e-12, 5.791e-12, 6.115e-12, 6.442e-12, 6.68e-12, 6.791e-12,
           6.831e-12, 6.839e-12, 6.946e-12, 7.128e-12, 7.537e-12, 8.036e-12, 8.392e-12, 8.526e-12, 8.11e-12, 7.325e-12, 6.329e-12, 5.183e-12,
02676
02677
           4.081e-12, 2.985e-12, 2.141e-12, 1.492e-12, 1.015e-12, 6.684e-13,
           4.414e-13, 2.987e-13, 2.038e-13, 1.391e-13, 9.86e-14, 7.24e-14,
02680
           5.493e-14, 4.288e-14, 3.427e-14, 2.787e-14, 2.296e-14, 1.909e-14,
           1.598e-14, 1.344e-14, 1.135e-14, 9.616e-15, 8.169e-15, 6.957e-15, 5.938e-15, 5.08e-15, 4.353e-15, 3.738e-15, 3.217e-15, 2.773e-15,
02681
02682
           2.397e-15, 2.077e-15, 1.805e-15, 1.575e-15, 1.382e-15, 1.221e-15, 1.09e-15, 9.855e-16, 9.068e-16, 8.537e-16, 8.27e-16, 8.29e-16,
02683
02684
           8.634e-16, 9.359e-16, 1.055e-15, 1.233e-15, 1.486e-15, 1.839e-15,
02685
02686
           2.326e-15, 2.998e-15, 3.934e-15, 5.256e-15, 7.164e-15, 9.984e-15,
02687
           1.427e-14, 2.099e-14, 3.196e-14, 5.121e-14, 7.908e-14, 1.131e-13,
           1.602e-13, 2.239e-13, 3.075e-13, 4.134e-13, 5.749e-13, 7.886e-13,
02688
           1.071e-12, 1.464e-12, 2.032e-12, 2.8e-12, 3.732e-12, 4.996e-12,
02689
           6.483e-12, 8.143e-12, 1.006e-11, 1.238e-11, 1.484e-11, 1.744e-11,
02690
           2.02e-11, 2.274e-11, 2.562e-11, 2.848e-11, 3.191e-11, 3.617e-11,
           4.081e-11, 4.577e-11, 4.937e-11, 5.204e-11, 5.401e-11, 5.462e-11,
02692
02693
           5.507e-11, 5.51e-11, 5.605e-11, 5.686e-11, 5.739e-11, 5.766e-11,
           5.74e-11, 5.754e-11, 5.761e-11, 5.777e-11, 5.712e-11, 5.51e-11,
02694
           5.088e-11, 4.438e-11, 3.728e-11, 2.994e-11, 2.305e-11, 1.715e-11, 1.256e-11, 9.208e-12, 6.745e-12, 5.014e-12, 3.785e-12, 2.9e-12,
02695
02696
           2.239e-12, 1.757e-12, 1.414e-12, 1.142e-12, 9.482e-13, 8.01e-13,
           6.961e-13, 6.253e-13, 5.735e-13, 5.433e-13, 5.352e-13, 5.493e-13,
02698
02699
           5.706e-13, 6.068e-13, 6.531e-13, 7.109e-13, 7.767e-13, 8.59e-13,
02700
           9.792e-13, 1.142e-12, 1.371e-12, 1.65e-12, 1.957e-12, 2.302e-12,
02701
           2.705e-12, 3.145e-12, 3.608e-12, 4.071e-12, 4.602e-12, 5.133e-12,
           5.572e-12, 5.987e-12, 6.248e-12, 6.533e-12, 6.757e-12, 6.935e-12,
02702
02703
           7.224e-12, 7.422e-12, 7.538e-12, 7.547e-12, 7.495e-12, 7.543e-12,
02704
           7.725e-12, 8.139e-12, 8.627e-12, 9.146e-12, 9.443e-12, 9.318e-12,
02705
           8.649e-12, 7.512e-12, 6.261e-12, 4.915e-12, 3.647e-12, 2.597e-12,
           1.785e-12, 1.242e-12, 8.66e-13, 6.207e-13, 4.61e-13, 3.444e-13, 2.634e-13, 2.1e-13, 1.725e-13, 1.455e-13, 1.237e-13, 1.085e-13,
02706
02707
           9.513e-14, 7.978e-14, 6.603e-14, 5.288e-14, 4.084e-14, 2.952e-14,
02708
           2.157e-14, 1.593e-14, 1.199e-14, 9.267e-15, 7.365e-15, 6.004e-15,
02710
           4.995e-15, 4.218e-15, 3.601e-15, 3.101e-15, 2.692e-15, 2.36e-15,
02711
           2.094e-15, 1.891e-15, 1.755e-15, 1.699e-15, 1.755e-15, 1.987e-15,
02712
           2.506e-15, 3.506e-15, 5.289e-15, 8.311e-15, 1.325e-14, 2.129e-14,
           3.237e-14, 4.595e-14, 6.441e-14, 8.433e-14, 1.074e-13, 1.383e-13,
02713
           1.762e-13, 2.281e-13, 2.831e-13, 3.523e-13, 4.38e-13, 5.304e-13, 6.29e-13, 7.142e-13, 8.032e-13, 8.934e-13, 9.888e-13, 1.109e-12,
02714
           1.261e-12, 1.462e-12, 1.74e-12, 2.099e-12, 2.535e-12, 3.008e-12,
02716
02717
           3.462e-12, 3.856e-12, 4.098e-12, 4.239e-12, 4.234e-12, 4.132e-12,
02718
           3.986e-12, 3.866e-12, 3.829e-12, 3.742e-12, 3.705e-12, 3.694e-12,
02719
           3.765e-12, 3.849e-12, 3.929e-12, 4.056e-12, 4.092e-12, 4.047e-12,
           3.792e-12, 3.407e-12, 2.953e-12, 2.429e-12, 1.931e-12, 1.46e-12,
02720
           1.099e-12, 8.199e-13, 6.077e-13, 4.449e-13, 3.359e-13, 2.524e-13,
02721
           1.881e-13, 1.391e-13, 1.02e-13, 7.544e-14, 5.555e-14, 4.22e-14,
           3.321e-14, 2.686e-14, 2.212e-14, 1.78e-14, 1.369e-14, 1.094e-14,
02723
02724
           9.13e-15, 8.101e-15, 7.828e-15, 8.393e-15, 1.012e-14, 1.259e-14,
           1.538e-14, 1.961e-14, 2.619e-14, 3.679e-14, 5.049e-14, 6.917e-14, 8.88e-14, 1.115e-13, 1.373e-13, 1.619e-13, 1.878e-13, 2.111e-13,
02725
02726
           2.33e-13, 2.503e-13, 2.613e-13, 2.743e-13, 2.826e-13, 2.976e-13,
02727
           3.162e-13, 3.36e-13, 3.491e-13, 3.541e-13, 3.595e-13, 3.608e-13,
02729
           3.709e-13, 3.869e-13, 4.12e-13, 4.366e-13, 4.504e-13, 4.379e-13,
02730
           3.955e-13, 3.385e-13, 2.741e-13, 2.089e-13, 1.427e-13, 9.294e-14,
02731
           5.775e-14, 3.565e-14, 2.21e-14, 1.398e-14, 9.194e-15, 6.363e-15,
           4.644e-15, 3.55e-15, 2.808e-15, 2.274e-15, 1.871e-15, 1.557e-15,
02732
           1.308e-15, 1.108e-15, 9.488e-16, 8.222e-16, 7.238e-16, 6.506e-16, 6.008e-16, 5.742e-16, 5.724e-16, 5.991e-16, 6.625e-16, 7.775e-16,
02733
02734
           9.734e-16, 1.306e-15, 1.88e-15, 2.879e-15, 4.616e-15, 7.579e-15,
02735
02736
           1.248e-14, 2.03e-14, 3.244e-14, 5.171e-14, 7.394e-14, 9.676e-14,
           1.199e-13, 1.467e-13, 1.737e-13, 2.02e-13, 2.425e-13, 3.016e-13, 3.7e-13, 4.617e-13, 5.949e-13, 7.473e-13, 9.378e-13, 1.191e-12,
02737
02738
           1.481e-12, 1.813e-12, 2.232e-12, 2.722e-12, 3.254e-12, 3.845e-12, 4.458e-12, 5.048e-12, 5.511e-12, 5.898e-12, 6.204e-12, 6.293e-12,
02739
02740
           6.386e-12, 6.467e-12, 6.507e-12, 6.466e-12, 6.443e-12, 6.598e-12,
02742
           6.873e-12, 7.3e-12, 7.816e-12, 8.368e-12, 8.643e-12, 8.466e-12,
02743
           7.871e-12, 6.853e-12, 5.714e-12, 4.482e-12, 3.392e-12, 2.613e-12,
           2.008e-12, 1.562e-12, 1.228e-12, 9.888e-13, 7.646e-13, 5.769e-13, 4.368e-13, 3.324e-13, 2.508e-13, 1.916e-13
02744
02745
02746
02747
02748
         static double xfcrev[15] =
02749
             1.003, 1.009, 1.015, 1.023, 1.029, 1.033, 1.037,
02750
           1.039, 1.04, 1.046, 1.036, 1.027, 1.01, 1.002, 1.
02751
02752
        double a1, a2, a3, dw, ew, dx, xw, xx, vf2, vf6, cw260, cw296,
02754
           sfac, fscal, cwfrn, ctmpth, ctwfrn, ctwslf;
02755
02756
        int iw, jw, ix;
02757
02758
        /* Get H2O continuum absorption... */
```

```
xw = nu / 10 + 1;
02760
                         if (xw >= 1 && xw < 2001) {
02761
                               iw = (int) xw;
                                jw = iw + 1;

dw = xw - iw;
02762
02763
                                ew = 1 - dw;
02764
                                cw296 = ew * h2o296[iw - 1] + dw * h2o296[jw - 1];
cw260 = ew * h2o260[iw - 1] + dw * h2o260[jw - 1];
cwfrn = ew * h2ofrn[iw - 1] + dw * h2ofrn[jw - 1];
02765
02766
02767
02768
                                if (nu <= 820 || nu >= 960) {
                                    sfac = 1;
02769
02770
                                } else {
                                 xx = (nu - 820) / 10;
02771
02772
                                       ix = (int) xx;
02773
                                       dx = xx - ix;
02774
                                    sfac = (1 - dx) * xfcrev[ix] + dx * xfcrev[ix + 1];
02775
02776
                                ctwslf = sfac * cw296 * pow(cw260 / cw296, (296 - t) / (296 - 260));
                                vf2 = gsl_pow_2(nu - 370);
02778
                                vf6 = gsl_pow_3(vf2);
                                fscal = 36100 / (vf2 + vf6 * 1e-8 + 36100) * -.25 + 1;
ctwfrn = cwfrn * fscal;
02779
02780
                                a1 = nu * u * tanh(.7193876 / t * nu);

a2 = 296 / t;

a3 = p / P0 * (q * ctwslf + (1 - q) * ctwfrn) * 1e-20;
02781
02782
02783
02784
                                ctmpth = a1 * a2 * a3;
02785
02786
                              ctmpth = 0;
02787
                          return ctmpth;
02788 }
02789
02791
02792 double ctmn2(
                         double nu,
02793
                         double p,
02794
02795
                         double t) {
02796
02797
                         static double ba[98] = \{ 0., 4.45e-8, 5.22e-8, 6.46e-8, 7.75e-8, 9.03e-8, 6.4e-8, 7.75e-8, 9.03e-8, 9.03
                              1.06e-7, 1.21e-7, 1.37e-7, 1.57e-7, 1.75e-7, 2.01e-7, 2.3e-7, 2.59e-7, 2.95e-7, 3.26e-7, 3.66e-7, 4.05e-7, 4.47e-7, 4.92e-7,
02798
02799
02800
                                5.34e-7, 5.84e-7, 6.24e-7, 6.67e-7, 7.14e-7, 7.26e-7, 7.54e-7,
                                7.84e-7, 8.09e-7, 8.42e-7, 8.62e-7, 8.87e-7, 9.11e-7, 9.36e-7,
02801
02802
                                9.76e-7, 1.03e-6, 1.11e-6, 1.23e-6, 1.39e-6, 1.61e-6, 1.76e-6,
                                1.94e-6, 1.97e-6, 1.87e-6, 1.75e-6, 1.56e-6, 1.42e-6, 1.35e-6,
02803
02804
                                1.32e-6, 1.29e-6, 1.29e-6, 1.3e-6, 1.32e-6, 1.33e-6,
                               1.32e 6, 1.35e 6, 1.33e 6, 1.31e 6, 1.32e 6, 1.32e 6, 1.32e 6, 1.34e 6, 1.35e 7, 1.35e 8, 1.3
02805
02806
02807
02808
02810
                                7.56e-8, 6.85e-8, 6.14e-8, 5.83e-8, 5.77e-8, 5e-8, 4.32e-8, 0.
02811
02812
                         static double betaa[98] = { 802., 802., 761., 722., 679., 646., 609., 562.,
511., 472., 436., 406., 377., 355., 338., 319., 299., 278., 255.,
233., 208., 184., 149., 107., 66., 25., -13., -49., -82., -104.,
02813
02814
                               233., 208., 184., 149., 107., 66., 25., -13., -49., -82., -104., -119., -130., -139., -144., -146., -146., -147., -148., -150., -153., -160., -169., -181., -189., -195., -200., -205., -209., -211., -210., -210., -209., -205., -199., -190., -180., -168., -157., -143., -126., -108., -89., -63., -32., 1., 35., 65., 95., 121., 141., 152., 161., 164., 164., 161., 155., 148., 143., 137., 133., 131., 133., 139., 150., 165., 187., 213., 248., 284., 321., 372., 449., 514., 569., 609., 642., 673., 673.
02816
02817
02818
02819
02820
02821
02822
02823
02824
                         static double nua[98] = { 2120., 2125., 2130., 2135., 2140., 2145., 2150.,
2155., 2160., 2165., 2170., 2175., 2180., 2185., 2190., 2195.,
2200., 2205., 2210., 2215., 2220., 2225., 2230., 2235., 2240.,
2245., 2250., 2255., 2260., 2265., 2270., 2275., 2280., 2285.,
02825
02826
02827
02829
                                 2290., 2295., 2300., 2305., 2310., 2315., 2320., 2325., 2330.,
02830
                                2335., 2340., 2345., 2350., 2355., 2360., 2365., 2370., 2375.,
02831
                                2380., 2385., 2390., 2395., 2400., 2405., 2410., 2415., 2420.,
02832
                                2425., 2430., 2435., 2440., 2445., 2450., 2455., 2460., 2465.,
                                2470., 2475., 2480., 2485., 2490., 2495., 2500., 2505., 2510., 2515., 2520., 2525., 2530., 2535., 2540., 2545., 2550., 2555., 2560., 2565., 2570., 2575., 2580., 2585., 2590., 2595., 2600., 2605.
02833
02834
02835
02836
02837
                         double b, beta, q_n2 = 0.79, t0 = 273, tr = 296;
02838
02839
02840
                         int idx:
02841
                          /* Check wavenumber range... */
02842
02843
                          if (nu < nua[0] || nu > nua[97])
02844
                               return 0;
02845
```

```
/* Interpolate B and beta... */
            idx = locate(nua, 98, nu);
02847
02848
           b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02849
           beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02850
02851
            /* Compute absorption coefficient... */
           return 0.1 * gsl_pow_2(p / P0) * gsl_pow_2(t0 / t) 
 * exp(beta * (1 / tr - 1 / t))
             * exp(beta * (1 / tr
02853
02854
               * q_n2 * b * (q_n2 + (1 - q_n2) * (1.294 - 0.4545 * t / tr));
02855 }
02856
02858
02859 double ctmo2(
02860
           double nu,
02861
           double p,
02862
           double t) {
02863
02864
            static double ba[90] = { 0., .061, .074, .084, .096, .12, .162, .208, .246,
              .285, .314, .38, .444, .5, .571, .673, .768, .853, .966, 1.097, 1.214, 1.333, 1.466, 1.591, 1.693, 1.796, 1.922, 2.037, 2.154, 2.264, 2.375, 2.508, 2.671, 2.847, 3.066, 3.417, 3.828, 4.204,
02866
02867
02868
               4.453, 4.599, 4.528, 4.284, 3.955, 3.678, 3.477, 3.346, 3.29,
              3.251, 3.231, 3.226, 3.212, 3.192, 3.108, 3.033, 2.911, 2.798, 2.646, 2.508, 2.322, 2.13, 1.928, 1.757, 1.588, 1.417, 1.253, 1.109, .99, .888, .791, .678, .587, .524, .464, .403, .357, .32, .29, .267, .242, .215, .182, .16, .146, .128, .103, .087, .081,
02869
02870
02871
02872
02873
               .071, .064, 0.
02874
02875
           static double betaa[90] = { 467., 467., 400., 315., 379., 368., 475., 521., 531., 512., 442., 444., 430., 381., 335., 324., 296., 248., 215., 193., 158., 127., 101., 71., 31., -6., -26., -47., -63., -79., -88., -88., -87., -90., -98., -99., -109., -134., -160., -167., -164., -158., -153., -151., -156., -166., -168., -173., -170., -161., -145., -126., -108., -84., -59., -29., 4., 41., 73., 97., 123., 159., 198., 220., 242., 256., 281., 311., 334., 319., 313., 321., 323., 310., 315., 320., 335., 361., 378., 373., 338., 319., 346., 322., 291., 290., 350., 371., 504., 504.
02876
02878
02879
02880
02881
02882
02884
02885
02886
           static double nua[90] = { 1360., 1365., 1370., 1375., 1380., 1385., 1390., 1395., 1400., 1405., 1410., 1415., 1420., 1425., 1430., 1435., 1440., 1445., 1450., 1455., 1460., 1465., 1470., 1475., 1480.,
02887
02888
02889
               1485., 1490., 1495., 1500., 1505., 1510., 1515., 1520., 1525., 1530., 1535., 1540., 1545., 1550., 1555., 1560., 1565., 1570.,
02891
02892
               1575., 1580., 1585., 1590., 1595., 1600., 1605., 1610., 1615.,
02893
               1620., 1625., 1630., 1635., 1640., 1645., 1650., 1655., 1660.,
               1665., 1670., 1675., 1680., 1685., 1690., 1695., 1700., 1705., 1710., 1715., 1720., 1725., 1730., 1735., 1740., 1745., 1750., 1755., 1760., 1765., 1770., 1775., 1780., 1785., 1790., 1795.,
02894
02895
02897
               1800., 1805.
02898
02899
           double b, beta, q_02 = 0.21, t0 = 273, tr = 296;
02900
02901
02902
02903
02904
            /* Check wavenumber range... */
02905
            if (nu < nua[0] || nu > nua[89])
02906
             return 0:
02907
02908
            /* Interpolate B and beta... */
02909
            idx = locate(nua, 90, nu);
02910
           b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02911
           beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02912
02913
            /* Compute absorption coefficient... */
           return 0.1 * gsl_pow_2(p / P0) * gsl_pow_2(t0 / t)  
* exp(beta * (1 / tr - 1 / t)) * q_o2 * b;
02914
02916 }
02917
02919
02920 void copy_atm(
           ctl_t * ctl,
atm_t * atm_dest,
02921
02922
02923
            atm_t * atm_src,
02924
           int init) {
02925
02926
           int ig, ip, iw;
02927
02928
02929
02930
            /* Data size... */
           s = (size_t) atm_src->np * sizeof(double);
02931
02932
```

```
/* Copy data... */
02934
       atm_dest->np = atm_src->np;
02935
       memcpy(atm_dest->time, atm_src->time, s);
02936
       memcpy(atm\_dest->z, atm\_src->z, s);
       memcpy(atm_dest->lon, atm_src->lon, s);
02937
02938
       memcpy(atm_dest->lat, atm_src->lat, s);
       memcpy(atm_dest->p, atm_src->p, s);
02939
02940
       memcpy(atm_dest->t, atm_src->t, s);
02941
       for (ig = 0; ig < ctl->ng; ig++)
02942
         memcpy(atm_dest->q[ig], atm_src->q[ig], s);
02943
       for (iw = 0; iw < ctl->nw; iw++)
02944
         memcpy(atm_dest->k[iw], atm_src->k[iw], s);
02945
02946
       /* Initialize... */
02947
       if (init)
02948
        for (ip = 0; ip < atm_dest->np; ip++) {
02949
           atm\_dest->p[ip] = 0;
           atm_dest->t[ip] = 0;
02950
           for (ig = 0; ig < ctl->ng; ig++)
02952
             atm_dest->q[ig][ip] = 0;
02953
           for (iw = 0; iw < ctl->nw; iw++)
02954
             atm_dest->k[iw][ip] = 0;
02955
02956 }
02957
02958 /
02959
02960 void copy_obs(
       ctl_t * ctl,
obs_t * obs_dest,
02961
02962
02963
       obs_t * obs_src,
02964
       int init) {
02965
02966
       int id, ir;
02967
02968
       size_t s;
02969
02970
       /* Data size... */
02971
       s = (size_t) obs_src->nr * sizeof(double);
02972
02973
       /* Copy data... */
       obs_dest->nr = obs_src->nr;
02974
02975
       memcpy(obs_dest->time, obs_src->time, s);
02976
       memcpy(obs_dest->obsz, obs_src->obsz, s);
02977
       memcpy(obs_dest->obslon, obs_src->obslon, s);
02978
       memcpy(obs_dest->obslat, obs_src->obslat, s);
02979
       memcpy(obs_dest->vpz, obs_src->vpz, s);
02980
       memcpy(obs_dest->vplon, obs_src->vplon, s);
       memcpy(obs_dest->vplat, obs_src->vplat, s);
02981
02982
       memcpy(obs_dest->tpz, obs_src->tpz, s);
       memcpy(obs_dest->tplon, obs_src->tplon, s);
02983
02984
       memcpy(obs_dest->tplat, obs_src->tplat, s);
02985
       for (id = 0; id < ctl->nd; id++)
       memcpy(obs_dest->rad[id], obs_src->rad[id], s);
for (id = 0; id < ctl->nd; id++)
02986
02987
         memcpy(obs_dest->tau[id], obs_src->tau[id], s);
02988
02989
02990
       /* Initialize... */
02991
       if (init)
02992
         for (id = 0; id < ctl->nd; id++)
           for (ir = 0; ir < obs_dest->nr; ir++)
  if (gsl_finite(obs_dest->rad[id][ir])) {
02993
02994
02995
              obs_dest->rad[id][ir] = 0;
02996
               obs_dest->tau[id][ir] = 0;
02997
02998 }
02999
03001
03002 int find_emitter(
03003
      ctl_t * ctl,
03004
       const char *emitter) {
03005
03006
       int ia:
03007
03008
       for (ig = 0; ig < ctl->ng; ig++)
03009
        if (strcasecmp(ctl->emitter[ig], emitter) == 0)
03010
          return ig;
03011
03012
       return -1:
03013 }
03014
03016
03017 void formod(
03018 ctl_t * ctl,
03019
       atm_t * atm,
```

```
obs_t * obs) {
03021
03022
        int id, ir, *mask;
03023
       /* Allocate... */
ALLOC(mask, int,
03024
03025
03026
              ND * NR);
03027
        /* Save observation mask... */
03028
        for (id = 0; id < ctl->nd; id++)
  for (ir = 0; ir < obs->nr; ir++)
03029
03030
           mask[id * NR + ir] = !gsl_finite(obs->rad[id][ir]);
03031
03032
03033
        /* Hydrostatic equilibrium... */
03034
       hydrostatic(ctl, atm);
03035
03036
        /* Claculate pencil beams... */
       for (ir = 0; ir < obs->nr; ir++)
  formod_pencil(ctl, atm, obs, ir);
03037
03038
03039
03040
        /* Apply field-of-view convolution... */
03041
        formod_fov(ctl, obs);
03042
03043
        /* Convert radiance to brightness temperature... */
03044
        if (ctl->write_bbt)
         for (id = 0; id < ctl->nd; id++)
03045
03046
            for (ir = 0; ir < obs->nr; ir++)
03047
              obs->rad[id][ir] = brightness(obs->rad[id][ir], ctl->nu[id]);
03048
03049
        /* Apply observation mask... */
       for (id = 0; id < ctl->nd; id+)
  for (ir = 0; ir < obs->nr; ir++)
03050
03051
03052
           if (mask[id * NR + ir])
03053
              obs->rad[id][ir] = GSL_NAN;
03054
       /* Free... */
03055
03056
       free(mask);
03057 }
03058
03060
03061 void formod continua(
       ctl_t * ctl,
los_t * los,
03062
03063
       int ip,
03064
03065
       double *beta) {
03066
       static int ig_co2 = -999, ig_h2o = -999;
03067
03068
03069
       int id:
03070
        /* Extinction... */
03071
        for (id = 0; id < ctl->nd; id++)
03072
03073
         beta[id] = los->k[ctl->window[id]][ip];
03074
03075
        /* CO2 continuum... */
03076
       if (ctl->ctm_co2) {
03077
         if (ig_co2 == -999)
03078
           ig_co2 = find_emitter(ctl, "CO2");
          03079
03080
03081
03082
                                 los->u[ig_co2][ip]) / los->ds[ip];
03083
03084
        /* H2O continuum... */
03085
03086
        if (ctl->ctm_h2o) {
         if (ig_h2o == -999)
03087
03088
           iq_h2o = find_emitter(ctl, "H2O");
03089
          if (ig_h2o >= 0)
03090
           for (id = 0; id < ctl->nd; id++)
03091
             beta[id] += ctmh2o(ctl->nu[id], los->p[ip], los->t[ip],
                                 los->q[ig_h2o][ip],
03092
                                 los->u[ig_h2o][ip]) / los->ds[ip];
03093
03094
03095
03096
        /* N2 continuum... */
03097
       if (ctl->ctm_n2)
         for (id = 0; id < ctl->nd; id++)
  beta[id] += ctmn2(ctl->nu[id], los->p[ip], los->t[ip]);
03098
03099
03100
03101
        /* 02 continuum... */
03102
        if (ctl->ctm_o2)
03103
          for (id = 0; id < ctl->nd; id++)
03104
           beta[id] += ctmo2(ctl->nu[id], los->p[ip], los->t[ip]);
03105 }
03106
```

```
03108
03109 void formod_fov(
        ctl_t * ctl,
obs_t * obs) {
03110
0.3111
03112
03113
        static double dz[NSHAPE], w[NSHAPE];
03114
03115
        static int init = 0, n;
0.3116
03117
        obs t *obs2;
03118
03119
        double rad[ND][NR], tau[ND][NR], wsum, z[NR], zfov;
03120
03121
        int i, id, idx, ir, ir2, nz;
03122
        /\star Do not take into account FOV... \star/
03123
        if (ctl->fov[0] == '-')
03124
03125
          return;
03126
03127
         /* Initialize FOV data... */
03128
        if (!init) {
        init = 1;
03129
          read_shape(ctl->fov, dz, w, &n);
03130
03131
03132
         /* Allocate... */
03133
03134
       ALLOC(obs2, obs_t, 1);
0.3135
03136
        /* Copy observation data... */
03137
        copy_obs(ctl, obs2, obs, 0);
03138
03139
        /* Loop over ray paths... */
for (ir = 0; ir < obs->nr; ir++) {
03140
03141
03142
          /* Get radiance and transmittance profiles... */
03143
          nz = 0;
03144
          for (ir2 = GSL_MAX(ir - NFOV, 0); ir2 < GSL_MIN(ir + 1 + NFOV, obs->nr);
               ir2++)
03145
03146
             if (obs->time[ir2] == obs->time[ir]) {
03147
              z[nz] = obs2->vpz[ir2];
              for (id = 0; id < ctl->nd; id++) {
  rad[id][nz] = obs2->rad[id][ir2];
0.3148
03149
                tau[id][nz] = obs2->tau[id][ir2];
03150
03151
03152
              nz++;
03153
          if (nz < 2)
03154
            ERRMSG("Cannot apply FOV convolution!");
03155
03156
03157
          /* Convolute profiles with FOV... */
03158
          for (id = 0; id < ctl->nd; id++) {
03159
            obs->rad[id][ir] = 0;
obs->tau[id][ir] = 0;
0.3160
03161
03162
03163
          for (i = 0; i < n; i++) {</pre>
03164
            zfov = obs->vpz[ir] + dz[i];
03165
             idx = locate(z, nz, zfov);
            for (id = 0; id < ctl->nd; id++) {
  obs->rad[id][ir] += w[i]
03166
0.3167
              * LIN(z[idx], rad[id][idx], z[idx + 1], rad[id][idx + 1], zfov);
obs->tau[id][ir] += w[i]
03168
03169
03170
                * LIN(z[idx], tau[id][idx], z[idx + 1], tau[id][idx + 1], zfov);
03171
03172
            wsum += w[i];
03173
          for (id = 0; id < ctl->nd; id++) {
03174
03175
            obs->rad[id][ir] /= wsum;
            obs->tau[id][ir] /= wsum;
03176
03177
0.3178
0.3179
03180
        /* Free... */
03181
        free (obs2);
03182 }
03183
03185
03186 void formed pencil(
       ctl_t * ctl,
atm_t * atm,
03187
03188
03189
        obs_t * obs,
03190
        int ir) {
03191
03192
        static tbl_t *tbl;
03193
```

```
static int init = 0;
03195
03196
      los_t *los;
03197
0.3198
      double beta_ctm[ND], eps, src_planck[ND], tau_path[NG][ND], tau_gas[ND];
03199
03200
       int id, ip;
03201
03202
       /\star Initialize look-up tables... \star/
03203
       if (!init) {
03204
        init = 1;
        ALLOC(tbl, tbl_t, 1);
03205
03206
        init_tbl(ctl, tbl);
03207
03208
03209
       /* Allocate... */
03210
      ALLOC(los, los_t, 1);
03211
03212
      /* Initialize... */
03213
       for (id = 0; id < ctl->nd; id++) {
03214
       obs->rad[id][ir] = 0;
03215
        obs->tau[id][ir] = 1;
03216
03217
03218
       /* Raytracing... */
      raytrace(ctl, atm, obs, los, ir);
03219
03220
03221
       /* Loop over LOS points... */
03222
       for (ip = 0; ip < los->np; ip++) {
03223
03224
         /* Get trace gas transmittance... */
03225
         intpol_tbl(ctl, tbl, los, ip, tau_path, tau_gas);
03226
03227
         /\star Get continuum absorption... \star/
03228
         formod_continua(ctl, los, ip, beta_ctm);
03229
03230
         /* Compute Planck function... */
         formod_srcfunc(ctl, tbl, los->t[ip], src_planck);
03231
03232
03233
         /* Loop over channels... */
03234
         for (id = 0; id < ctl->nd; id++)
          if (tau_gas[id] > 0) {
03235
03236
03237
            /* Get segment emissivity... */
03238
            eps = 1 - tau_gas[id] * exp(-beta_ctm[id] * los->ds[ip]);
03239
03240
            /* Compute radiance... */
03241
            obs->rad[id][ir] += src_planck[id] * eps * obs->tau[id][ir];
03242
03243
             /* Compute path transmittance... */
03244
            obs->tau[id][ir] *= (1 - eps);
03245
03246
03247
       /* Add surface... */
03248
03249
       if (los->tsurf > 0) {
       formod_srcfunc(ctl, tbl, los->tsurf, src_planck);
03251
        for (id = 0; id < ctl->nd; id++)
03252
           obs->rad[id][ir] += src_planck[id] * obs->tau[id][ir];
03253
03254
03255
       /* Free... */
03256
      free(los);
03257 }
03258
03260
03261 void formod srcfunc(
03262
      ctl t * ctl.
      tbl_t * tbl,
03263
03264
      double t,
03265
      double *src) {
03266
03267
      int id, it;
03268
03269
      /* Determine index in temperature array... */
03270
      it = locate(tbl->st, TBLNS, t);
03271
03272
       /\star Interpolate Planck function value... \star/
       for (id = 0; id < ctl->nd; id++)
03273
        03274
03275
03276 }
03277
03279
03280 void geo2cart(
```

```
03281
        double z,
03282
        double lon,
03283
        double lat,
03284
        double *x) {
03285
03286
        double radius;
03287
03288
        x[0] = radius * cos(lat / 180 * M_PI) * cos(lon / 180 * M_PI);

x[1] = radius * cos(lat / 180 * M_PI) * sin(lon / 180 * M_PI);

x[2] = radius * sin(lat / 180 * M_PI);
03289
03290
03291
03292 }
03293
03295
03296 double gravity(
03297
        double z
03298
        double lat) {
03299
03300
        /* Compute gravity according to 1967 Geodetic Reference System...
03301
        return 9.780318 * (1 + 0.0053024 * gsl_pow_2(sin(lat / 180 * M_PI))
03302
                              - 0.0000058 * gsl_pow_2(sin(2 * lat / 180 * M_PI))) -
           3.086e-3 * z:
03303
03304 }
03305
03307
03308 void hydrostatic(
03309
        ctl_t * ctl,
03310
        atm_t * atm) {
03311
03312
        static int ig_h2o = -999;
03313
03314
        double dzmin = 1e99, e = 0, mean, mmair = 28.96456e-3, mmh2o =
03315
          18.0153e-3, z;
03316
03317
        int i, ip, ipref = 0, ipts = 20;
03318
03319
        /* Check reference height... */
03320
        if (ctl->hydz < 0)
           return;
03321
03322
        /* Determine emitter index of H2O... */
03323
03324
        if (ig_h2o == -999)
          ig_h2o = find_emitter(ctl, "H2O");
03325
03326
03327
         /* Find air parcel next to reference height... */
        for (ip = 0; ip < atm->np; ip++)
  if (fabs(atm->z[ip] - ctl->hydz) < dzmin) {
    dzmin = fabs(atm->z[ip] - ctl->hydz);
03328
03329
03330
             ipref = ip;
03331
03332
03333
        /* Upper part of profile... */
for (ip = ipref + 1; ip < atm->np; ip++) {
03334
03335
           mean = 0;
03336
           for (i = 0; i < ipts; i++) {</pre>
03337
03338
             z = LIN(0.0, atm->z[ip-1], ipts-1.0, atm->z[ip], (double) i);
             if (ig_h2o >= 0)
03339
03340
               e = LIN(0.0, atm->q[ig_h2o][ip - 1],
             ipts - 1.0, atm \rightarrow q[ig_h2o][ip], (double) i); mean += (e * mmh2o + (1 - e) * mmair)
03341
03342
03343
               * gravity(z, atm->lat[ipref]) / GSL_CONST_MKSA_MOLAR_GAS
03344
                / LIN(0.0, atm->t[ip - 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03345
03346
03347
           /* Compute p(z,T) ... */
03348
           atm->p[ip] =
03349
             \exp(\log(atm - p[ip - 1]) - mean * 1000 * (atm - z[ip] - atm - z[ip - 1]));
03350
03351
03352
         /\star Lower part of profile... \star/
03353
         for (ip = ipref - 1; ip >= 0; ip--) {
          mean = 0;
for (i = 0; i < ipts; i++) {
03354
03355
03356
             z = LIN(0.0, atm \rightarrow z[ip + 1], ipts - 1.0, atm \rightarrow z[ip], (double) i);
03357
             if (ig_h2o >= 0)
03358
               e = LIN(0.0, atm->q[ig_h2o][ip + 1],
             ipts - 1.0, atm->q[ig_h20][ip], (double) i);
mean += (e * mmh2o + (1 - e) * mmair)
  * gravity(z, atm->lat[ipref]) / GSL_CONST_MKSA_MOLAR_GAS
  / LIN(0.0, atm->t[ip + 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03359
03360
03361
03362
03363
03364
03365
           /* Compute p(z,T)... */
03366
           atm->p[ip]
             \exp(\log(atm - p[ip + 1]) - mean * 1000 * (atm - z[ip] - atm - z[ip + 1]));
03367
```

```
03368
03369 }
03370
03372
03373 void idx2name(
03374
     ctl_t * ctl,
03375
       int idx,
03376
       char *quantity) {
03377
03378
       int iq, iw;
03379
03380
       if (idx == IDXP)
         sprintf(quantity, "PRESSURE");
03381
03382
03383
       if (idx == IDXT)
         sprintf(quantity, "TEMPERATURE");
03384
03385
03386
       for (ig = 0; ig < ctl->ng; ig++)
        if (idx == IDXQ(ig))
03387
           sprintf(quantity, "%s", ctl->emitter[ig]);
03388
03389
03390
       for (iw = 0; iw < ctl->nw; iw++)
         if (idx == IDXK(iw))
03391
           sprintf(quantity, "EXTINCT_WINDOW%d", iw);
03392
03393 }
03394
03396
03397 void init tbl(
03398
       ctl_t * ctl,
03399
       tbl_t * tbl) {
03400
03401
       FILE *in;
03402
       char filename[LEN], line[LEN];
03403
03404
03405
       double eps, eps_old, press, press_old, temp, temp_old, u, u_old,
03406
         f[NSHAPE], fsum, nu[NSHAPE];
03407
03408
       int i, id, ig, ip, it, n;
03409
       /* Loop over trace gases and channels... */
0.3410
03411 for (ig = 0; ig < ctl->ng; ig++)
03412 #pragma omp parallel for default (none) shared(ctl,tbl,ig) private(in,filename,line,eps,eps_old,press,
     press_old, temp, temp_old, u, u_old, id, ip, it)
03413
         for (id = 0; id < ctl->nd; id++) {
03414
            /* Initialize... */
03415
           tbl->np[ig][id] = -1;
eps_old = -999;
03416
03417
           press_old = -999;
temp_old = -999;
03418
03419
           u_old = -999;
03420
03421
            /* Try to open file... */
03422
           sprintf(filename, "%s_%.4f_%s.tab",
03423
03424
                   ctl->tblbase, ctl->nu[id], ctl->emitter[ig]);
03425
            if (!(in = fopen(filename, "r"))) {
             printf("Missing emissivity table: %s\n", filename);
03426
03427
             continue;
03428
03429
           printf("Read emissivity table: %s\n", filename);
03430
03431
            /* Read data... */
03432
           while (fgets(line, LEN, in)) {
03433
             /* Parse line... */
if (sscanf(line, "%lg %lg %lg %lg", &press, &temp, &u, &eps) != 4)
03434
03435
03436
               continue;
03437
03438
             /* Determine pressure index... */
             if (press != press_old) {
  press_old = press;
03439
03440
               if ((++tbl->np[ig][id]) >= TBLNP)
03441
03442
                 ERRMSG("Too many pressure levels!");
03443
               tbl->nt[ig][id][tbl->np[ig][id]] = -1;
03444
03445
03446
             /* Determine temperature index... */
             if (temp != temp_old) {
03447
               temp_old = temp;
03448
03449
               if ((++tbl->nt[ig][id][tbl->np[ig][id]]) >= TBLNT)
03450
                 ERRMSG("Too many temperatures!");
               tbl->nu[ig][id][tbl->np[ig][id]]
03451
03452
                 [tbl->nt[ig][id][tbl->np[ig][id]]] = -1;
03453
             }
```

```
/* Determine column density index... */    if ((eps > eps_old && u > u_old) || tbl->nu[ig][id][tbl->np[ig][id]]
03455
03456
                  [tbl->nt[ig][id][tbl->np[ig][id]]] < 0) {
03457
03458
                eps_old = eps;
                u_old = u;
03459
                if ((++tbl->nu[ig][id][tbl->np[ig][id]]
03460
03461
                      [tbl->nt[ig][id][tbl->np[ig][id]]]) >= TBLNU) {
03462
                  tbl->nu[ig][id][tbl->np[ig][id]]
03463
                    [tbl->nt[ig][id][tbl->np[ig][id]]]--;
03464
                  continue:
03465
03466
              }
03467
03468
              /* Store data... */
03469
              tbl \rightarrow p[ig][id][tbl \rightarrow np[ig][id]] = press;
              \label{tbl-} t[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
03470
03471
                = temp;
03472
              tbl->u[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
03473
                [tbl->nu[ig][id][tbl->np[ig][id]]
03474
                  [tbl->nt[ig][id][tbl->np[ig][id]]] = (float) u;
03475
              [tbl->nu[ig][id][tbl->np[ig][id]]
03476
                  [tbl->nt[ig][id][tbl->np[ig][id]]] = (float) eps;
03477
03478
03479
03480
            /* Increment counters... */
03481
            tbl->np[ig][id]++;
03482
            for (ip = 0; ip < tbl->np[ig][id]; ip++) {
              tbl->nt[ig][id][ip]++;
for (it = 0; it < tbl->nt[ig][id][ip]; it++)
03483
03484
03485
                tbl->nu[ig][id][ip][it]++;
03486
03487
03488
            /* Close file... */
03489
            fclose(in);
03490
          }
03491
03492
        /* Write info... */
03493
       printf("Initialize source function table...\n");
03494
        /* Loop over channels... */
03495
03496 #pragma omp parallel for default(none) shared(ctl,tbl,ig) private(filename,it,i,n,f,fsum,nu)
        for (id = 0; id < ctl->nd; id++) {
03497
03498
03499
          /* Read filter function...
03500
          sprintf(filename, "%s_%.4f.filt", ctl->tblbase, ctl->nu[id]);
03501
          read_shape(filename, nu, f, &n);
03502
03503
          /* Compute source function table... */
          for (it = 0; it < TBLNS; it++) {</pre>
03504
03505
03506
            /* Set temperature... */
03507
            tbl->st[it] = LIN(0.0, TMIN, TBLNS - 1.0, TMAX, (double) it);
03508
03509
            /* Integrate Planck function... */
            fsum = 0;
03510
03511
            tbl \rightarrow sr[id][it] = 0;
03512
            for (i = 0; i < n; i++) {
03513
              fsum += f[i];
03514
              tbl->sr[id][it] += f[i] * planck(tbl->st[it], nu[i]);
03515
03516
            tbl->sr[id][it] /= fsum;
03517
03518
       }
03519 }
03520
03521 /
       ******************************
03522
03523 void intpol_atm(
       ctl_t * ctl,
atm_t * atm,
03524
03525
03526
        double z,
03527
        double *p,
03528
        double *t,
03529
        double *q,
03530
       double *k) {
03531
03532
       int ig, ip, iw;
03533
        /* Get array index... */
03534
03535
        ip = locate(atm->z, atm->np, z);
03536
03537
       *p = EXP(atm->z[ip], atm->p[ip], atm->z[ip + 1], atm->p[ip + 1], z);
*t = LIN(atm->z[ip], atm->t[ip], atm->z[ip + 1], atm->t[ip + 1], z);
03538
03539
        for (ig = 0; ig < ctl->ng; ig++)
03540
```

```
q[ig] =
03542
           LIN(atm->z[ip], atm->q[ig][ip], atm->z[ip+1], atm->q[ig][ip+1], z);
03543
        for (iw = 0; iw < ctl->nw; iw++)
03544
         k[iw] =
03545
           LIN(atm->z[ip], atm->k[iw][ip], atm->z[ip + 1], atm->k[iw][ip + 1], z);
03546 }
03547
03549
03550 void intpol_tbl(
03551
       ctl_t * ctl,
       tbl_t * tbl,
03552
        los_t * los,
03553
03554
       int ip,
03555
       double tau_path[NG][ND],
03556
       double tau_seg[ND]) {
03557
03558
       double eps, eps00, eps01, eps10, eps11, u;
03559
03560
       int id, ig, ipr, it0, it1;
03561
03562
       /* Initialize... */
03563
       if (ip <= 0)</pre>
         for (ig = 0; ig < ctl->ng; ig++)
  for (id = 0; id < ctl->nd; id++)
03564
03565
             tau_path[ig][id] = 1;
03566
03567
03568
        /* Loop over channels... */
03569
       for (id = 0; id < ctl->nd; id++) {
03570
03571
          /* Initialize... */
03572
         tau_seg[id] = 1;
03573
03574
          /\star Loop over emitters.... \star/
03575
          for (ig = 0; ig < ctl->ng; ig++) {
03576
03577
            /* Check size of table (pressure)... */
03578
           if (tbl->np[ig][id] < 2)</pre>
03579
             eps = 0;
03580
03581
            /* Check transmittance... */
           else if (tau_path[ig][id] < 1e-9)</pre>
03582
             eps = 1;
03583
03584
03585
            /* Interpolate... */
03586
03587
03588
              /\star Determine pressure and temperature indices... \star/
              ipr = locate(tbl->p[ig][id], tbl->np[ig][id], los->p[ip]);
it0 = locate(tbl->t[ig][id][ipr], tbl->nt[ig][id][ipr], los->t[ip]);
03589
03590
             it1 =
03591
03592
               locate(tbl->t[ig][id][ipr + 1], tbl->nt[ig][id][ipr + 1],
03593
                       los->t[ip]);
03594
              03595
03596
03597
03598
                  || tbl->nu[ig][id][ipr][it0 + 1] < 2
                  || tbl->nu[ig][id][ipr + 1][it1] < 2
|| tbl->nu[ig][id][ipr + 1][it1 + 1] < 2)
03599
03600
03601
                eps = 0:
03602
03603
              else {
03604
                /\star Get emissivities of extended path... \star/
03605
                u = intpol_tbl_u(tbl, ig, id, ipr, it0, 1 - tau_path[ig][id]);
03606
03607
                eps00 = intpol\_tbl\_eps(tbl, ig, id, ipr, it0, u + los->u[ig][ip]);
03608
03609
               u = intpol_tbl_u(tbl, iq, id, ipr, it0 + 1, 1 - tau_path[iq][id]);
               eps01 =
03610
03611
                 intpol_tbl_eps(tbl, ig, id, ipr, it0 + 1, u + los->u[ig][ip]);
03612
03613
                u = intpol_tbl_u(tbl, ig, id, ipr + 1, it1, 1 - tau_path[ig][id]);
03614
                eps10 =
03615
                  intpol tbl eps(tbl, iq, id, ipr + 1, it1, u + los->u[iq][ip]);
03616
03617
03618
                 intpol_tbl_u(tbl, ig, id, ipr + 1, it1 + 1, 1 - tau_path[ig][id]);
                eps11 =
03619
                  intpol tbl eps(tbl, ig, id, ipr + 1, it1 + 1, u + los->
03620
     u[ig][ip]);
03621
03622
                /* Interpolate with respect to temperature... */
03623
                eps00 = LIN(tbl->t[ig][id][ipr][it0], eps00,
                           tbl->t[ig][id][ipr][it0 + 1], eps01, los->t[ip]);
03624
                03625
03626
```

```
03628
              /* Interpolate with respect to pressure... */
               03629
03630
03631
               /* Check emssivity range... */
03632
               eps00 = GSL_MAX(GSL_MIN(eps00, 1), 0);
03633
03634
03635
               /\star Determine segment emissivity..
03636
               eps = 1 - (1 - eps00) / tau_path[ig][id];
             }
03637
03638
03639
03640
           /* Get transmittance of extended path... */
03641
           tau_path[ig][id] *= (1 - eps);
03642
           /* Get segment transmittance... */
tau_seg[id] *= (1 - eps);
03643
03644
03645
03646
       }
03647 }
03648
03649 /
       03650
03651 double intpol_tbl_eps(
03652
       tbl_t * tbl,
03653
       int ig,
03654
       int id,
03655
       int ip,
03656
       int it,
03657
       double u) {
03658
03659
       int idx;
03660
       /* Lower boundary... */
if (u < tbl->u[ig][id][ip][it][0])
03661
03662
        return LIN(0, 0, tbl->u[ig][id][ip][it][0], tbl->eps[ig][id][ip][it][0],
03663
03664
                    u);
03665
03666
       /* Upper boundary... */
03667
       else if (u > tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
         return LIN(tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03668
                    03669
03670
                    1e30, 1, u);
03671
03672
       /* Interpolation... */
03673
       else {
03674
03675
         /* Get index... */
         idx = locate_tbl(tbl->u[iq][id][ip][it], tbl->nu[iq][id][ip][it], u);
03676
03677
03678
03679
         return
            \begin{split} & LIN(tbl->u[ig][id][ip][it][idx], & tbl->eps[ig][id][ip][it][idx], \\ & tbl->u[ig][id][ip][it][idx + 1], & tbl->eps[ig][id][ip][it][idx + 1], \end{split} 
03680
03681
03682
               u);
03683
       }
03684 }
03685
03687
03688 double intpol_tbl_u(
03689
       tbl_t * tbl,
03690
       int ig,
03691
       int id,
03692
       int ip,
03693
       int it,
03694
       double eps) {
03695
03696
       int idx;
03697
03698
       /* Lower boundary... */
03699
       if (eps < tbl->eps[ig][id][ip][it][0])
         return LIN(0, 0, tbl->eps[ig][id][ip][it][0], tbl->u[ig][id][ip][it][0],
03700
03701
                    eps);
03702
03703
       /* Upper boundary... */
03704
       else if (eps > tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
03705
         return LIN(tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03706
                    tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03707
                    1, 1e30, eps);
03708
03709
       /* Interpolation... */
03710
       else {
03711
03712
         /* Get index... */
03713
         idx = locate_tbl(tbl->eps[iq][id][ip][it], tbl->nu[iq][id][ip][it], eps);
```

```
03714
03715
                    /* Interpolate... */
03716
                      LIN(tbl->eps[ig][id][ip][it][idx], tbl->u[ig][id][ip][it][idx], tbl->eps[ig][id][ip][it][idx + 1], tbl->u[ig][id][ip][it][idx + 1],
0.3717
03718
03719
                                eps);
03720
03721 }
03722
03724
03725 void jsec2time(
03726
               double isec,
03727
               int *year,
03728
                int *mon,
03729
               int *day,
03730
               int *hour.
03731
               int *min,
03732
               int *sec,
03733
               double *remain) {
03734
03735
               struct tm t0, *t1;
03736
03737
               time_t jsec0;
03738
03739
               t0.tm_year = 100;
03740
                t0.tm_mon = 0;
03741
                t0.tm_mday = 1;
               t0.tm\_hour = 0;
03742
               t0.tm_min = 0;
03743
03744
               t0.tm sec = 0;
03745
03746
               jsec0 = (time_t) jsec + timegm(&t0);
03747
               t1 = gmtime(&jsec0);
03748
03749
               *year = t1->tm_year + 1900;
               *mon = t1->tm_mon + 1;
*day = t1->tm_mday;
03750
03751
03752
                *hour = t1->tm_hour;
03753
               *min = t1->tm_min;
               *sec = t1->tm_sec;
03754
03755
               *remain = jsec - floor(jsec);
03756 }
03757
03759
03760 void kernel(
03761
               ctl_t * ctl,
03762
               atm_t * atm,
               obs_t * obs,
03763
03764
               gsl_matrix * k) {
03765
03766
               atm_t *atm1;
03767
               obs_t *obs1;
03768
03769
               gsl_vector *x0, *x1, *yy0, *yy1;
03770
03771
               int *iqa, j;
03772
03773
               double h;
03774
03775
               size t i, n, m;
03776
03777
               /* Get sizes... */
03778
               m = k -> size1;
03779
               n = k->size2;
03780
03781
                /* Allocate... */
03782
               x0 = gsl_vector_alloc(n);
03783
                yy0 = gsl_vector_alloc(m);
03784
                ALLOC(iqa, int,
                           N);
03785
03786
03787
                /\star Compute radiance for undisturbed atmospheric data... \star/
03788
               formod(ctl, atm, obs);
03789
03790
               /* Compose vectors... */
03791
               atm2x(ctl, atm, x0, iqa, NULL);
03792
               obs2y(ctl, obs, yy0, NULL, NULL);
03793
03794
               /* Initialize kernel matrix... */
03795
               gsl_matrix_set_zero(k);
03796
03797
               /* Loop over state vector elements... */
03798 \text{ \#pragma omp parallel for default(none) shared(ctl,atm,obs,k,x0,yy0,n,m,iqa) private(i, j, h, x1, yy1, atm1, b, k, x1, yy1, atm1, b, x1, yy1, atm1, b
              obs1)
03799
               for (j = 0; j < (int) n; j++) {
```

```
03801
          /* Allocate... */
03802
         x1 = gsl_vector_alloc(n);
03803
         yy1 = gsl_vector_alloc(m);
         ALLOC(atm1, atm_t, 1);
ALLOC(obs1, obs_t, 1);
03804
03805
03806
03807
          /\star Set perturbation size... \star/
03808
          if (iqa[j] == IDXP)
           h = GSL_MAX(fabs(0.01 * gsl_vector_get(x0, (size_t) j)), 1e-7);
03809
03810
         else if (iqa[j] == IDXT)
03811
           h = 1;
         else if (iqa[j] >= IDXQ(0) && iqa[j] < IDXQ(ctl->ng))
03812
03813
           h = GSL_MAX(fabs(0.01 * gsl_vector_get(x0, (size_t) j)), 1e-15);
03814
          else if (iqa[j] \geq= IDXK(0) && iqa[j] \leq IDXK(ctl-\geqnw))
03815
           h = 1e-4;
03816
          else
03817
           ERRMSG("Cannot set perturbation size!");
03818
03819
          /* Disturb state vector element... */
03820
          gsl_vector_memcpy(x1, x0);
03821
          gsl_vector_set(x1, (size_t) j, gsl_vector_get(x1, (size_t) j) + h);
         copy_atm(ctl, atml, atm, 0);
copy_obs(ctl, obs1, obs, 0);
x2atm(ctl, x1, atml);
03822
03823
03824
03825
03826
          /\star Compute radiance for disturbed atmospheric data... \star/
03827
          formod(ctl, atm1, obs1);
03828
03829
          /\star Compose measurement vector for disturbed radiance data... \star/
03830
         obs2y(ctl, obs1, yy1, NULL, NULL);
03831
03832
          /* Compute derivatives... */
          for (i = 0; i < m; i++)</pre>
03833
03834
           gsl_matrix_set(k, i, (size_t) j,
                          (gsl_vector_get(yy1, i) - gsl_vector_get(yy0, i)) / h);
03835
03836
         /* Free... */
03837
03838
         gsl_vector_free(x1);
03839
          gsl_vector_free(yy1);
03840
          free (atm1);
03841
         free (obs1);
03842
03843
03844
        /* Free... */
03845
       gsl_vector_free(x0);
03846
       gsl_vector_free(yy0);
03847
       free(iqa);
03848 }
03849
03851
03852 int locate(
03853
       double *xx,
03854
       int n,
03855
       double x) {
03856
03857
       int i, ilo, ihi;
03858
03859
       ilo = 0;
       ihi = n - 1;
03860
       i = (ihi + ilo) >> 1;
03861
03862
03863
       if (xx[i] < xx[i + 1])
03864
         while (ihi > ilo + 1) {
           i = (ihi + ilo) >> 1;
03865
03866
           if (xx[i] > x)
             ihi = i;
03867
03868
           else
03869
             ilo = i;
03870
       } else
03871
         while (ihi > ilo + 1) {
           i = (ihi + ilo) >> 1;
03872
           if (xx[i] <= x)</pre>
03873
             ihi = i;
03874
03875
           else
03876
             ilo = i;
03877
         }
03878
03879
       return ilo:
03880 }
03881
03883
03884 int locate_tbl(
03885 float *xx,
03886
       int n.
```

```
03887
       double x) {
03888
03889
       int i, ilo, ihi;
03890
       ilo = 0;
03891
03892
       ihi = n - 1;
       i = (ihi + ilo) >> 1;
03893
03894
       while (ihi > ilo + 1) {
  i = (ihi + ilo) >> 1;
03895
03896
         if (xx[i] > x)
03897
          ihi = i;
03898
03899
         else
03900
          ilo = i;
03901
       }
03902
03903
       return ilo:
03904 }
03905
03907
03908 size_t obs2y(
       ctl_t * ctl,
obs_t * obs,
03909
03910
03911
       qsl_vector * y,
03912
       int *ida,
03913
       int *ira) {
03914
03915
       int id, ir;
03916
03917
       size t m = 0:
03918
03919
       /* Determine measurement vector... */
03920
       for (ir = 0; ir < obs->nr; ir++)
        for (id = 0; id < ctl->nd; id++)
  if (gsl_finite(obs->rad[id][ir])) {
03921
03922
            if (y != NULL)
    gsl_vector_set(y, m, obs->rad[id][ir]);
03923
03924
03925
             if (ida != NULL)
03926
              ida[m] = id;
03927
             if (ira != NULL)
              ira[m] = ir;
03928
03929
             m++;
03930
03931
03932
       return m;
03933 }
03934
03936
03937 double planck(
03938 double t,
03939
       double nu)
03940
03941
       return C1 * gsl_pow_3(nu) / gsl_expm1(C2 * nu / t);
03942 }
03943
03945
03946 void raytrace(
03947
       ctl_t * ctl,
atm_t * atm,
03948
       obs_t * obs,
los_t * los,
03949
03950
03951
       int ir) {
03952
03953
       double cosa, d, dmax, dmin = 0, ds, ex0[3], ex1[3], frac, h = 0.02, k[NW],
03954
        lat, lon, n, naux, ng[3], norm, p, q[NG], t, x[3], xh[3],
         xobs[3], xvp[3], z = 1e99, zmax, zmin, zrefrac = 60;
03955
03956
03957
       int i, ig, ip, iw, stop = 0;
03958
03959
       /* Initialize... */
03960
       los->np = 0;
los->tsurf = -999;
03961
03962
       obs->tpz[ir] = obs->vpz[ir];
03963
       obs->tplon[ir] = obs->vplon[ir];
03964
       obs->tplat[ir] = obs->vplat[ir];
03965
03966
       /* Get altitude range of atmospheric data... */
03967
       gsl_stats_minmax(&zmin, &zmax, atm->z, 1, (size_t) atm->np);
03968
       /* Check observer altitude... */
if (obs->obsz[ir] < zmin)</pre>
03969
03970
03971
         ERRMSG("Observer below surface!");
03972
03973
       /* Check view point altitude... */
```

```
03974
         if (obs->vpz[ir] > zmax)
03975
03976
03977
         /\star Determine Cartesian coordinates for observer and view point... \star/
03978
         geo2cart(obs->obsz[ir], obs->obslon[ir], obs->obslat[ir], xobs);
geo2cart(obs->vpz[ir], obs->vplon[ir], obs->vplat[ir], xvp);
03979
03980
03981
          /* Determine initial tangent vector... */
         for (i = 0; i < 3; i++)
  ex0[i] = xvp[i] - xobs[i];</pre>
03982
03983
         norm = NORM(ex0);

for (i = 0; i < 3; i++)

ex0[i] /= norm;
03984
03985
03986
03987
03988
         /\star Observer within atmosphere... \star/
         for (i = 0; i < 3; i++)
x[i] = xobs[i];</pre>
03989
03990
03991
03992
         /* Observer above atmosphere (search entry point)... */
03993
         if (obs->obsz[ir] > zmax) {
03994
           dmax = norm;
03995
            while (fabs(dmin - dmax) > 0.001) {
             d = (dmax + dmin) / 2;
for (i = 0; i < 3; i++)
x[i] = xobs[i] + d * ex0[i];</pre>
03996
03997
03998
03999
              cart2geo(x, &z, &lon, &lat);
04000
              if (z \le zmax && z > zmax - 0.001)
04001
                break;
              if (z < zmax - 0.0005)
04002
               dmax = d;
04003
04004
              else
04005
                dmin = d;
04006
04007
04008
         /* Ray-tracing... */
04009
04010
         while (1) {
04011
04012
            /* Set step length... */
04013
            ds = ctl->rayds;
            if (ctl->raydz > 0) {
04014
              norm = NORM(x);
04015
              for (i = 0; i < 3; i++)</pre>
04016
                xh[i] = x[i] / norm;
04017
04018
              cosa = fabs(DOTP(ex0, xh));
04019
              if (cosa != 0)
04020
                ds = GSL_MIN(ctl->rayds, ctl->raydz / cosa);
04021
04022
04023
            /* Determine geolocation... */
            cart2geo(x, &z, &lon, &lat);
04025
04026
            /\star Check if LOS hits the ground or has left atmosphere... \star/
            if (z < zmin || z > zmax) {
  stop = (z < zmin ? 2 : 1);</pre>
04027
04028
04029
              frac =
04030
                ((z <
04031
                   zmin ? zmin : zmax) - los->z[los->np - 1]) / (z - los->z[los->np -
04032
04033
              geo2cart(los->z[los->np-1], los->lon[los->np-1],
              los->lat[los->np - 1], xh);
for (i = 0; i < 3; i++)
04034
04035
04036
               x[i] = xh[i] + frac * (x[i] - xh[i]);
04037
              cart2geo(x, &z, &lon, &lat);
04038
              los \rightarrow ds[los \rightarrow np - 1] = ds * frac;
04039
              ds = 0;
04040
04041
04042
            /* Interpolate atmospheric data... */
04043
            intpol_atm(ctl, atm, z, &p, &t, q, k);
04044
04045
            /* Save data... */
04046
            los -> lon[los -> np] = lon;
            los->lat[los->np] = lat;
04047
            los->z[los->np] = z;
los->p[los->np] = p;
04048
04049
04050
            los->t[los->np] = t;
04051
            for (ig = 0; ig < ctl->ng; ig++)
04052
              los \rightarrow q[ig][los \rightarrow np] = q[ig];
            for (iw = 0; iw < ctl->nw; iw++)
los->k[iw][los->np] = k[iw];
04053
04054
04055
            los->ds[los->np] = ds;
04056
04057
            /* Increment and check number of LOS points... */
            if ((++los->np) > NLOS)
   ERRMSG("Too many LOS points!");
04058
04059
04060
```

```
04061
           /* Check stop flag... */
04062
           if (stop) {
04063
             los - > tsurf = (stop == 2 ? t : -999);
04064
             break;
04065
04066
04067
           /* Determine refractivity... */
04068
           if (ctl->refrac && z <= zrefrac)</pre>
04069
            n = 1 + refractivity(p, t);
04070
           else
04071
            n = 1;
04072
           /* Construct new tangent vector (first term)... */ for (i = 0; i < 3; i++)
04073
04074
04075
             ex1[i] = ex0[i] * n;
04076
           /* Compute gradient of refractivity... */
04077
04078
           if (ctl->refrac && z <= zrefrac) {</pre>
             for (i = 0; i < 3; i++)
04080
               xh[i] = x[i] + 0.5 * ds * ex0[i];
04081
             cart2geo(xh, &z, &lon, &lat);
04082
             intpol_atm(ctl, atm, z, &p, &t, q, k);
             n = refractivity(p, t);
for (i = 0; i < 3; i++) {
   xh[i] += h;</pre>
04083
04084
04085
04086
               cart2geo(xh, &z, &lon, &lat);
04087
               intpol_atm(ctl, atm, z, &p, &t, q, k);
               naux = refractivity(p, t);
ng[i] = (naux - n) / h;
04088
04089
               xh[i] -= h;
04090
04091
04092
           } else
04093
             for (i = 0; i < 3; i++)</pre>
04094
               ng[i] = 0;
04095
04096
           /* Construct new tangent vector (second term)... */
           for (i = 0; i < 3; i++)
  ex1[i] += ds * ng[i];</pre>
04097
04098
04099
04100
           /* Normalize new tangent vector... */
           norm = NORM(ex1);
for (i = 0; i < 3; i++)
  ex1[i] /= norm;</pre>
04101
04102
04103
04104
04105
           /\star Determine next point of LOS... \star/
04106
           for (i = 0; i < 3; i++)
04107
             x[i] += 0.5 * ds * (ex0[i] + ex1[i]);
04108
04109
           /* Copy tangent vector... */
for (i = 0; i < 3; i++)</pre>
04110
             ex0[i] = ex1[i];
04111
04112
04113
04114
         /* Get tangent point (to be done before changing segment lengths!)... */
04115
        tangent_point(los, &obs->tpz[ir], &obs->tplon[ir], &obs->
      tplat[ir]);
04116
04117
         /* Change segment lengths according to trapezoid rule... */
        for (ip = los->np - 1; ip >= 1; ip--)
los->ds[ip] = 0.5 * (los->ds[ip - 1] + los->ds[ip]);
04118
04119
        los->ds[0] *= 0.5;
04120
04121
04122
         /* Compute column density... */
04123
         for (ip = 0; ip < los->np; ip++)
04124
           for (ig = 0; ig < ctl->ng; ig++)
04125
             los -> u[ig][ip] = 10 * los -> q[ig][ip] * los -> p[ip]
04126
               / (GSL_CONST_MKSA_BOLTZMANN * los->t[ip]) * los->ds[ip];
04127 }
04128
04130
04131 void read_atm(
        const char *dirname,
const char *filename,
04132
04133
        ctl_t * ctl,
04134
04135
        atm_t * atm) {
04136
04137
        FILE *in;
04138
        char file[LEN], line[LEN], *tok;
04139
04140
04141
        int ig, iw;
04142
04143
         /* Init... */
04144
        atm->np = 0;
04145
04146
         /* Set filename... */
```

```
04147
         if (dirname != NULL)
04148
            sprintf(file, "%s/%s", dirname, filename);
04149
          else
04150
            sprintf(file, "%s", filename);
04151
04152
          /* Write info... */
         printf("Read atmospheric data: %s\n", file);
04153
04154
04155
          /* Open file... */
          if (!(in = fopen(file, "r")))
04156
            ERRMSG("Cannot open file!");
04157
04158
04159
          /* Read line... */
04160
         while (fgets(line, LEN, in)) {
04161
            /* Read data... */
TOK(line, tok, "%lg", atm->time[atm->np]);
TOK(NULL, tok, "%lg", atm->z[atm->np]);
TOK(NULL, tok, "%lg", atm->lon[atm->np]);
04162
04163
04164
04165
            TOK (NULL, tok, "%1g", atm->lon[atm->np]);
TOK (NULL, tok, "%1g", atm->lat[atm->np]);
TOK (NULL, tok, "%1g", atm->p[atm->np]);
TOK (NULL, tok, "%1g", atm->t[atm->np]);
for (ig = 0; ig < ctl->ng; ig++)
TOK (NULL, tok, "%1g", atm->q[ig][atm->np]);
for (iw = 0; iw < ctl->nw; iw++)
TOK (NULL, tok, "%1g", atm->k[iw][atm->np]);
04166
04167
04168
04169
04170
04171
04172
04173
            /* Increment data point counter... */
if ((++atm->np) > NP)
04174
04175
               ERRMSG("Too many data points!");
04176
04177
04178
04179
          /* Close file... */
04180
          fclose(in);
04181
04182
          /* Check number of points... */
04183
          if (atm->np < 1)
04184
             ERRMSG("Could not read any data!");
04185 }
04186
04188
04189 void read ctl(
04190
         int argc,
         char *argv[],
04191
04192
         ctl_t * ctl) {
04193
04194
         int id, ig, iw;
04195
04196
         /* Write info... */
         printf("\nJuelich Rapid Spectral Simulation Code (JURASSIC)\n"
04197
                   "(executable: %s | compiled: %s, %s)\n\n",
04198
04199
                   argv[0], __DATE__, __TIME__);
04200
04201
          /* Emitters... */
          ctl->ng = (int) scan_ctl(argc, argv, "NG", -1, "0", NULL);
04202
             (ctl->ng < 0 || ctl->ng > NG)
04203
04204
            ERRMSG("Set 0 <= NG <= MAX!");</pre>
         for (ig = 0; ig < ctl->ng; ig++)
    scan_ctl(argc, argv, "EMITTER", ig, "", ctl->emitter[ig]);
04205
04206
04207
04208
         /* Radiance channels... */
04209
          ctl->nd = (int) scan_ctl(argc, argv, "ND", -1, "0", NULL);
04210
          if (ctl->nd < 0 || ctl->nd > ND)
04211
            ERRMSG("Set 0 <= ND <= MAX!");</pre>
04212
          for (id = 0; id < ctl->nd; id++)
            ctl->nu[id] = scan_ctl(argc, argv, "NU", id, "", NULL);
04213
04214
04215
          /* Spectral windows... */
          ctl->nw = (int) scan_ctl(argc, argv, "NW", -1, "1", NULL);
04216
04217
          if (ctl->nw < 0 || ctl->nw > NW)
04218
            ERRMSG("Set 0 <= NW <= MAX!");
04219
          for (id = 0; id < ctl->nd; id++)
            ctl->window[id] = (int) scan_ctl(argc, argv, "WINDOW", id, "0", NULL);
04220
04221
         /* Emissivity look-up tables... */
scan_ctl(argc, argv, "TBLBASE", -1, "-", ctl->tblbase);
04222
04223
04224
04225
          /* Hydrostatic equilibrium... */
         ctl->hydz = scan_ctl(argc, argv, "HYDZ", -1, "-999", NULL);
04226
04227
04228
          /* Continua... */
         /* Continua... */
ctl->ctm_co2 = (int) scan_ctl(argc, argv, "CTM_CO2", -1, "1", NULL);
ctl->ctm_h2o = (int) scan_ctl(argc, argv, "CTM_H2O", -1, "1", NULL);
ctl->ctm_n2 = (int) scan_ctl(argc, argv, "CTM_N2", -1, "1", NULL);
ctl->ctm_o2 = (int) scan_ctl(argc, argv, "CTM_O2", -1, "1", NULL);
04229
04230
04231
04232
04233
```

```
/* Ray-tracing... */
        ctl->refrac = (int) scan_ctl(argc, argv, "REFRAC", -1, "1", NULL);
ctl->rayds = scan_ctl(argc, argv, "RAYDS", -1, "10", NULL);
ctl->raydz = scan_ctl(argc, argv, "RAYDZ", -1, "0.5", NULL);
04235
04236
04237
04238
        /* Field of view... */
scan_ctl(argc, argv, "FOV", -1, "-", ctl->fov);
04239
04240
04241
         /* Retrieval interface... */
04242
        ctl->retp_zmin = scan_ctl(argc, argv, "RETP_ZMIN", -1, "-999", NULL);
ctl->retp_zmax = scan_ctl(argc, argv, "RETP_ZMAX", -1, "-999", NULL);
ctl->rett_zmin = scan_ctl(argc, argv, "RETT_ZMIN", -1, "-999", NULL);
ctl->rett_zmax = scan_ctl(argc, argv, "RETT_ZMAX", -1, "-999", NULL);
04243
04244
04245
04246
04247
         for (ig = 0; ig < ctl->ng; ig++) {
          ctl->retq_zmin[ig] = scan_ctl(argc, argv, "RETO_ZMIN", ig, "-999", NULL); ctl->retq_zmax[ig] = scan_ctl(argc, argv, "RETO_ZMAX", ig, "-999", NULL);
04248
04249
04250
04251
        for (iw = 0; iw < ctl->nw; iw++) {
         ctl->retk_zmin[iw] = scan_ctl(argc, argv, "RETK_ZMIN", iw, "-999", NULL);
04252
           ctl->retk_zmax[iw] = scan_ctl(argc, argv, "RETK_ZMAX", iw, "-999", NULL);
04253
04254
04255
04256
        /* Output flags... */
        ctl->write_bbt = (int) scan_ctl(argc, argv, "WRITE_BBT", -1, "0", NULL);
04257
04258
        ctl->write_matrix =
04259
          (int) scan_ctl(argc, argv, "WRITE_MATRIX", -1, "0", NULL);
04260 }
04261
04263
04264 void read matrix(
        const char *dirname,
const char *filename,
04265
04266
04267
        gsl_matrix * matrix) {
04268
        FILE *in;
04269
04270
04271
        char dum[LEN], file[LEN], line[LEN];
04272
04273
        double value;
04274
04275
        int i, j;
04276
04277
         /* Set filename... */
04278
         if (dirname != NULL)
04279
           sprintf(file, "%s/%s", dirname, filename);
04280
          sprintf(file, "%s", filename);
04281
04282
04283
        /* Write info... */
        printf("Read matrix: %s\n", file);
04284
04285
04286
         /* Open file... */
04287
        if (!(in = fopen(file, "r")))
           ERRMSG("Cannot open file!");
04288
04289
04290
        /* Read data... */
04291
        gsl_matrix_set_zero(matrix);
        04292
04293
             &i, dum, dum, dum, dum, dum, &j, dum, dum, dum, dum, dum, dum, dum, &value) == 13)
gsl_matrix_set(matrix, (size_t) i, (size_t) j, value);
04294
04295
04296
04297
04298
        /* Close file... */
04299
        fclose(in);
04300 }
04301
04303
04304 void read_obs(
04305
        const char *dirname,
04306
        const char *filename,
04307
        ctl_t * ctl,
04308
        obs_t * obs) {
04309
04310
        FILE *in;
04311
04312
        char file[LEN], line[LEN], *tok;
04313
04314
        int id;
04315
04316
         /* Init... */
04317
        obs->nr = 0;
04318
        /* Set filename... */
if (dirname != NULL)
04319
04320
```

```
04321
            sprintf(file, "%s/%s", dirname, filename);
04322
            sprintf(file, "%s", filename);
04323
04324
04325
         /* Write info... */
04326
         printf("Read observation data: %s\n", file);
04327
04328
04329
         if (!(in = fopen(file, "r")))
            ERRMSG("Cannot open file!");
04330
04331
04332
         /* Read line... */
04333
         while (fgets(line, LEN, in)) {
04334
             /* Read data...
04335
            TOK(line, tok, "%lg", obs->time[obs->nr]);
TOK(NULL, tok, "%lg", obs->obsz[obs->nr]);
TOK(NULL, tok, "%lg", obs->obslon[obs->nr]);
TOK(NULL, tok, "%lg", obs->obslat[obs->nr]);
04336
04337
04338
04339
            TOK(NULL, tok, "%lg", obs->obslat[obs->nr]);
TOK(NULL, tok, "%lg", obs->vpz[obs->nr]);
TOK(NULL, tok, "%lg", obs->vplon[obs->nr]);
TOK(NULL, tok, "%lg", obs->vplat[obs->nr]);
TOK(NULL, tok, "%lg", obs->tpz[obs->nr]);
TOK(NULL, tok, "%lg", obs->tplon[obs->nr]);
TOK(NULL, tok, "%lg", obs->tplon[obs->nr]);
TOK(NULL, tok, "%lg", obs->tplat[obs->nr]);
for (id = 0; id < ctl->nd; id++)
    TOK(NULL, tok, "%lg", obs->tau[id][obs->nr]);
04340
04341
04342
04343
04344
04345
04346
04347
04348
04349
04350
04351
            /* Increment counter... */
04352
            if ((++obs->nr) > NR)
04353
              ERRMSG("Too many rays!");
04354
04355
         /* Close file... */
04356
04357
         fclose(in);
04358
04359
          /* Check number of points... */
04360
         if (obs->nr < 1)
04361
            ERRMSG("Could not read any data!");
04362 }
04363
04365
04366 void read_shape(
04367
         const char *filename,
04368
         double *x,
04369
         double *v.
04370
         int *n) {
04371
04372
         FILE *in;
04373
04374
         char line[LEN];
04375
04376
         /* Write info... */
04377
         printf("Read shape function: %s\n", filename);
04378
04379
          /* Open file... *,
         if (!(in = fopen(filename, "r")))
04380
            ERRMSG("Cannot open file!");
04381
04382
04383
         /* Read data... */
04384
         *n = 0;
04385
         while (fgets(line, LEN, in))
           if (sscanf(line, "%lg %lg", &x[*n], &y[*n]) == 2)
   if ((++(*n)) > NSHAPE)
        ERRMSG("Too many data points!");
04386
04387
04388
04389
04390
         /* Check number of points... */
04391
04392
            ERRMSG("Could not read any data!");
04393
         /* Close file... */
04394
04395
         fclose(in);
04396 }
04397
04399
04400 double refractivity(
04401 double p,
04402
         double t) {
04403
04404
         /\star Refractivity of air at 4 to 15 micron... \star/
04405
         return 7.753e-05 * p / t;
04406 }
04407
```

```
04409
04410 double scan_ctl(
04411
        int argc,
04412
        char *argv[],
const char *varname,
04413
04414
        int arridx,
04415
        const char *defvalue,
04416
        char *value) {
04417
04418
        FILE *in = NULL:
04419
        char dummy[LEN], fullname1[LEN], fullname2[LEN], line[LEN],
  msg[LEN], rvarname[LEN], rval[LEN];
04420
04421
04422
04423
        int contain = 0, i;
04424
04425
         /* Open file... */
        if (argv[1][0] != '-')
04426
         if
              (!(in = fopen(argv[1], "r")))
04427
04428
             ERRMSG("Cannot open file!");
04429
04430
        /* Set full variable name... */
        if (arridx >= 0) {
04431
          sprintf(fullname1, "%s[%d]", varname, arridx);
sprintf(fullname2, "%s[*]", varname);
04432
04433
04434
         sprintf(fullname1, "%s", varname);
sprintf(fullname2, "%s", varname);
04435
04436
04437
04438
04439
        /* Read data... */
04440
        if (in != NULL)
04441
          while (fgets(line, LEN, in))
            if (sscanf(line, "%s %s %s", rvarname, dummy, rval) == 3)
if (strcasecmp(rvarname, fullname1) == 0 ||
04442
04443
                   strcasecmp(rvarname, fullname2) == 0) {
04444
04445
                 contain = 1;
04446
                 break;
04447
        for (i = 1; i < argc - 1; i++)
04448
         if (strcasecmp(argv[i], fullname1) == 0 ||
    strcasecmp(argv[i], fullname2) == 0) {
    sprintf(rval, "%s", argv[i + 1]);
04449
04450
04451
04452
             contain = 1;
04453
             break;
04454
          }
04455
        /* Close file... */
04456
        if (in != NULL)
04457
04458
          fclose(in);
04459
04460
        /* Check for missing variables... */
04461
        if (!contain) {
04462
          if (strlen(defvalue) > 0)
            sprintf(rval, "%s", defvalue);
04463
04464
           else {
04465
             sprintf(msg, "Missing variable %s!\n", fullname1);
04466
             ERRMSG(msg);
04467
04468
        }
04469
04470
        /* Write info... */
04471
        printf("%s = %s\n", fullname1, rval);
04472
04473
         /* Return values... */
        if (value != NULL)
  sprintf(value, "%s", rval);
04474
04475
04476
        return atof(rval);
04478
04480
04481 void tangent_point(
04482
        los_t * los,
double *tpz,
04483
04484
        double *tplon,
04485
        double *tplat) {
04486
        double a, b, c, dummy, v[3], v0[3], v2[3], x, x1, x2, yy0, yy1, yy2;
04487
04488
04489
        size_t i, ip;
04490
04491
         /* Find minimum altitude... */
04492
        ip = gsl_stats_min_index(los->z, 1, (size_t) los->np);
04493
04494
        /* Nadir or zenith... */
```

```
if (ip <= 0 || ip >= (size_t) los->np - 1) {
        *tpz = los->z[los->np - 1];
04496
         *tplon = los->lon[los->np - 1];
04497
         *tplat = los->lat[los->np - 1];
04498
04499
04500
04501
        /* Limb... */
04502
04503
04504
          /* Determine interpolating polynomial y=a*x^2+b*x+c... */
         yy0 = los \rightarrow z[ip - 1];
04505
         yy1 = los \rightarrow z[ip];
04506
04507
          yy2 = los -> z[ip + 1];
04508
          x1 = sqrt(gsl_pow_2(los->ds[ip]) - gsl_pow_2(yy1 - yy0));
04509
          x2 = x1 + sqrt(gsl_pow_2(los->ds[ip + 1]) - gsl_pow_2(yy2 - yy1));
          a = 1 / (x1 - x2) * (-(yy0 - yy1) / x1 + (yy0 - yy2) / x2);   b = -(yy0 - yy1) / x1 - a * x1; 
04510
04511
         c = yy0;
04512
04514
          /* Get tangent point location... */
04515
          x = -b / (2 * a);
          *tpz = a * x * x + b * x + c;
04516
          geo2cart(los->z[ip - 1], los->lon[ip - 1], los->lat[ip - 1], v0);
geo2cart(los->z[ip + 1], los->lon[ip + 1], los->lat[ip + 1], v2);
for (i = 0; i < 3; i++)</pre>
04517
04518
04519
04520
           v[i] = LIN(0.0, v0[i], x2, v2[i], x);
04521
         cart2geo(v, &dummy, tplon, tplat);
04522
04523 }
04524
04526
04527 void time2jsec(
       int year,
04528
04529
       int mon,
04530
       int day,
04531
       int hour,
       int min,
04533
       int sec,
04534
       double remain,
04535
       double *jsec) {
04536
04537
       struct tm t0, t1;
04538
04539
       t0.tm_year = 100;
04540
        t0.tm_mon = 0;
       t0.tm_mday = 1;
t0.tm_hour = 0;
04541
04542
       t0.tm_min = 0;
04543
04544
       t0.tm\_sec = 0;
04545
04546
        t1.tm_year = year - 1900;
04547
        t1.tm_mon = mon - 1;
       t1.tm_mday = day;
04548
       t1.tm_hour = hour;
04549
       t1.tm_min = min;
04550
04551
        t1.tm_sec = sec;
04552
04553
       *jsec = (double) timegm(&t1) - (double) timegm(&t0) + remain;
04554 }
04555
04557
04558 void timer(
04559
       const char *name,
04560
       const char *file,
04561
       const char *func,
04562
       int line.
04563
       int mode) {
04564
04565
       static double dt_w, w0[10];
04566
04567
       static int 10[10], nt;
04568
04569
       struct timeval tim;
04570
04571
        /* Start new timer... */
04572
        if (mode == 1) {
         gettimeofday(&tim, NULL);
04573
04574
         w0[nt] = (double) tim.tv_sec + (double) tim.tv_usec / 1e6;
10[nt] = line;
04575
             ((++nt) >= 10)
04577
            ERRMSG("Too many timers!");
04578
04579
       /* Write elapsed time... */
04580
04581
       else {
```

```
04582
04583
           /* Check timer index... */
04584
           if (nt - 1 < 0)
             ERRMSG("Coding error!");
04585
04586
04587
           /* Get time differences... */
04588
           gettimeofday(&tim, NULL);
04589
           dt_w = (double) tim.tv_sec + (double) tim.tv_usec / 1e6 - w0[nt - 1];
04590
           04591
04592
04593
04594
04595
04596
         /* Stop timer... */
04597
         if (mode == 3)
04598
           nt--:
04599 }
04600
04602
04603 void write_atm(
        const char *dirname,
const char *filename,
04604
04605
04606
        ctl_t * ctl,
        atm_t * atm)
04607
04608
04609
        FILE *out;
04610
04611
        char file[LEN];
04612
04613
        int iq, ip, iw, n = 6;
04614
04615
         /* Set filename... */
        if (dirname != NULL)
  sprintf(file, "%s/%s", dirname, filename);
04616
04617
04618
         else
04619
           sprintf(file, "%s", filename);
04620
04621
         /* Write info... */
04622
         printf("Write atmospheric data: %s\n", file);
04623
04624
         /* Create file... */
04625
         if (!(out = fopen(file, "w")))
04626
           ERRMSG("Cannot create file!");
04627
         /* Write header... */
04628
         04629
04630
                  "# $2 = altitude [km] \n"
04631
04632
                  "# $3 = longitude [deg]\n
04633
                  "# $4 = latitude [deg] \n"
                  "# $5 = pressure [hPa] \n" "# $6 = temperature [K] \n");
04634
        for (ig = 0; ig < ctl->ng; ig+)
    fprintf(out, "# $%d = %s volume mixing ratio\n", ++n, ctl->emitter[ig]);
for (iw = 0; iw < ctl->nw; iw++)
    fprintf(out, "# $%d = window %d: extinction [1/km]\n", ++n, iw);
04635
04636
04637
04638
04639
        /* Write data... */
for (ip = 0; ip < atm->np; ip++) {
   if (ip == 0 || atm->lat[ip] != atm->lat[ip - 1]
        || atm->lon[ip] != atm->lon[ip - 1])
        fprintf(out, "\n");
   fprintf(out, "\2.2f \%g \%g \%g \%g", atm->time[ip], atm->z[ip],
        atm->lon[ip], atm->lat[ip], atm->p[ip], atm->t[ip]);
   for (ig = 0) ig < ctl->ng: ig+)
04640
04641
04642
04643
04644
04645
04646
           for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, " %g", atm->q[ig][ip]);
04647
04648
           for (iw = 0; iw < ctl->nw; iw++)
04649
           fprintf(out, " %g", atm->k[iw][ip]);
fprintf(out, "\n");
04650
04651
04652
04653
04654
         /* Close file... */
04655
        fclose(out);
04656 }
04657
04659
04660 void write_matrix(
        const char *dirname,
const char *filename,
04661
04662
         ctl_t * ctl,
04663
04664
         gsl_matrix * matrix,
04665
         atm_t * atm,
04666
        obs_t * obs,
        const char *rowspace,
const char *colspace,
04667
04668
```

```
04669
       const char *sort) {
04670
04671
        FILE *out;
04672
04673
        char file[LEN], quantity[LEN];
04674
04675
        int *cida, *ciqa, *cipa, *cira, *rida, *riqa, *ripa, *rira;
04676
04677
        size_t i, j, nc, nr;
04678
        /* Check output flag... */
04679
        if (!ctl->write_matrix)
04680
04681
         return:
04682
        /* Allocate... */
04683
04684
        ALLOC(cida, int, M);
04685
        ALLOC(ciga, int,
04686
              N);
        ALLOC(cipa, int,
04687
04688
              N);
04689
        ALLOC(cira, int,
04690
              M);
        ALLOC(rida, int,
04691
04692
              M):
04693
        ALLOC(riga, int,
04694
              N);
04695
        ALLOC(ripa, int,
04696
             N);
        ALLOC(rira, int,
04697
04698
              M);
04699
04700
        /* Set filename...
04701
        if (dirname != NULL)
04702
         sprintf(file, "%s/%s", dirname, filename);
04703
        else
         sprintf(file, "%s", filename);
04704
04705
04706
        /* Write info... */
04707
        printf("Write matrix: %s\n", file);
04708
04709
        /* Create file... */
        if (!(out = fopen(file, "w")))
04710
         ERRMSG("Cannot create file!");
04711
04712
04713
        /* Write header (row space)... */
04714
        if (rowspace[0] == 'y') {
04715
          fprintf(out, "# $1 = Row: index (measurement space) n"
04716
04717
04718
                   "# $2 = Row: channel wavenumber [cm^-1]\n"
                   "# $3 = Row: time (seconds since 2000-01-01T00:00Z)\n"
04720
                   "# $4 = Row: view point altitude [km]\n"
04721
                   "# $5 = Row: view point longitude [deg] \n"
04722
                  "# $6 = Row: view point latitude [deg]\n");
04723
          /* Get number of rows... */
04724
04725
         nr = obs2y(ctl, obs, NULL, rida, rira);
04726
04727
        } else {
04728
          fprintf(out.
04729
04730
                   "# $1 = Row: index (state space) \n"
04731
                   "# $2 = Row: name of quantity\n"
04732
                   "# $3 = Row: time (seconds since 2000-01-01T00:00Z)\n"
04733
                   "# $4 = Row: altitude [km]\n"
                   "# $5 = Row: longitude [deg]\n" "# $6 = Row: latitude [deg]\n");
04734
04735
04736
          /* Get number of rows... */
04737
          nr = atm2x(ctl, atm, NULL, riga, ripa);
04738
04739
04740
        /* Write header (column space)... */
04741
        if (colspace[0] == 'y') {
04742
04743
          fprintf(out,
04744
                   "# \$7 = \text{Col: index (measurement space)} \n"
04745
                   "# $8 = Col: channel wavenumber [cm^-1]\n"
                   "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"    # $10 = Col: view point altitude [km]\n"
04746
04747
                   "# $11 = Col: view point longitude [deg]\n"
"# $12 = Col: view point latitude [deg]\n");
04748
04749
04750
04751
          /* Get number of columns... */
04752
          nc = obs2y(ctl, obs, NULL, cida, cira);
04753
04754
        } else {
04755
```

```
04756
          fprintf(out,
                    "# $7 = Col: index (state space) n"
04757
04758
                    "# $8 = Col: name of quantity n"
                    "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04759
04760
                    "# $10 = Col: altitude [km] \n"
                   "# $11 = Col: longitude [deg]\n" "# $12 = Col: latitude [deg]\n");
04761
04762
04763
           /\star Get number of columns... \star/
04764
          nc = atm2x(ctl, atm, NULL, ciqa, cipa);
04765
04766
        /* Write header entry... */
fprintf(out, "# $13 = Matrix element n'n);
04767
04768
04769
04770
        /* Write matrix data... */
        i = j = 0;
while (i < nr && j < nc) {
04771
04772
04773
04774
          /* Write info about the row... */
          if (rowspace[0] == 'y')
  fprintf(out, "%d %g %.2f %g %g %g",
04775
04776
04777
                      (int) i, ctl->nu[rida[i]],
04778
                      obs->time[rira[i]], obs->vpz[rira[i]],
04779
                      obs->vplon[rira[i]], obs->vplat[rira[i]]);
04780
          else {
04781
            idx2name(ct1, riqa[i], quantity);
fprintf(out, "%d %s % .2f %g %g %g", (int) i, quantity,
04782
04783
                     atm->time[ripa[i]], atm->z[ripa[i]],
04784
                      atm->lon[ripa[i]], atm->lat[ripa[i]]);
04785
          }
04786
04787
           /\star Write info about the column... \star/
          if (colspace[0] == 'y')
  fprintf(out, " %d %g %.2f %g %g %g",
04788
04789
                      (int) j, ctl->nu[cida[j]],
04790
                     obs->time[cira[j]], obs->vpz[cira[j]],
obs->vplon[cira[j]], obs->vplat[cira[j]]);
04791
04792
04793
             idx2name(ctl, ciqa[j], quantity);
fprintf(out, " %d %s %.2f %g %g %g", (int) j, quantity,
04794
04795
                      atm->time[cipa[j]], atm->z[cipa[j]]
04796
                     atm->lon[cipa[j]], atm->lat[cipa[j]]);
04797
04798
          }
04799
04800
           /\star Write matrix entry... \star/
04801
           fprintf(out, " %g\n", gsl_matrix_get(matrix, i, j));
04802
04803
           /\star Set matrix indices... \star/
04804
           if (sort[0] == 'r') {
04805
             j++;
             if (j >= nc) {
04806
04807
              j = 0;
04808
               i++:
04809
              fprintf(out, "\n");
04810
04811
          } else {
04812
            i++;
04813
             if (i >= nr) {
              i = 0;
04814
04815
               j++;
               fprintf(out, "\n");
04816
04817
             }
04818
          }
04819
04820
04821
        /* Close file... */
04822
        fclose(out);
04823
04824
        /* Free... */
04825
        free(cida);
04826
        free(ciqa);
04827
        free(cipa);
04828
        free(cira);
04829
        free (rida):
04830
        free (riga);
04831
        free (ripa);
04832
        free (rira);
04833 }
04834
04836
04837 void write_obs(
04838
       const char *dirname,
04839
        const char *filename,
        ctl_t * ctl,
obs_t * obs) {
04840
04841
04842
```

```
04843
       FILE *out;
04844
04845
        char file[LEN];
04846
04847
        int id, ir, n = 10;
04848
04849
        /* Set filename...
04850
        if (dirname != NULL)
04851
         sprintf(file, "%s/%s", dirname, filename);
04852
        else
         sprintf(file, "%s", filename);
04853
04854
04855
        /* Write info... */
04856
       printf("Write observation data: %s\n", file);
04857
04858
        /* Create file... */
        if (!(out = fopen(file, "w")))
04859
         ERRMSG("Cannot create file!");
04860
04861
04862
        /* Write header... */
04863
        fprintf(out,
04864
                "# $1 = time (seconds since 2000-01-01T00:00Z)\n"
                "# $2 = observer altitude [km] \n"
04865
                "# $3 = observer longitude [deg]\n'
04866
04867
                "# $4 = observer latitude [deg]\n'
                "# $5 = view point altitude [km]\n"
04868
04869
                "# $6 = view point longitude [deg]\n"
04870
                "# $7 = view point latitude [deg] n"
                "# $8 = tangent point altitude [km]\n"
04871
                "# $9 = tangent point longitude [deg]\n"
04872
04873
                "# $10 = tangent point latitude [deg]\n");
04874
        for (id = 0; id < ctl->nd; id++)
04875
        fprintf(out, "# \$%d = channel %g: radiance [W/(m^2 sr cm^-1)]\n",
04876
                  ++n, ctl->nu[id]);
       for (id = 0; id < ctl->nd; id++)
  fprintf(out, "# $%d = channel %g: transmittance\n", ++n, ctl->nu[id]);
04877
04878
04879
        /* Write data... */
04881
        for (ir = 0; ir < obs->nr; ir++) {
        if (ir == 0 || obs->time[ir] != obs->time[ir - 1])
   fprintf(out, "\n");
fprintf(out, "%.2f %g %g %g %g %g %g %g %g", obs->time[ir],
04882
04883
04884
                  obs->obsz[ir], obs->obslon[ir], obs->obslat[ir],
04885
                  obs->tpz[ir], obs->tplon[ir], obs->tplat[ir], obs->tpz[ir], obs->tplon[ir], obs->tplat[ir],
04886
04887
          for (id = 0; id < ctl->nd; id++)
  fprintf(out, " %g", obs->rad[id][ir]);
04888
04889
          for (id = 0; id < ctl->nd; id+)
  fprintf(out, " %g", obs->tau[id][ir]);
fprintf(out, "\n");
04890
04891
04892
04893
04894
04895
        /* Close file... */
04896
       fclose(out);
04897 }
04898
04900
04901 void x2atm(
04902
       ctl_t * ctl,
       gsl vector * x.
04903
04904
       atm t * atm) {
04905
04906
       int ig, iw;
04907
04908
       size_t n = 0;
04909
        /* Set pressure... */
04910
       x2atm_help(atm, ctl->retp_zmin, ctl->retp_zmax, atm->
04911
     p, x, &n);
04912
04913
        /* Set temperature... */
04914
       x2atm_help(atm, ctl->rett_zmin, ctl->rett_zmax, atm->
     t, x, &n);
04915
04916
        /* Set volume mixing ratio... */
04917
        for (ig = 0; ig < ctl->ng; ig++)
04918
        x2atm_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
04919
                     atm->q[ig], x, &n);
04920
04921
       /* Set extinction... */
        for (iw = 0; iw < ctl->nw; iw++)
04922
04923
         x2atm_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
04924
                     atm->k[iw], x, &n);
04925 }
04926
```

```
04928
04929 void x2atm_help(
04930
       atm_t * atm,
       double zmin,
04931
04932
       double zmax,
04933
       double *value.
04934
       gsl_vector * x,
04935
       size_t * n) {
04936
04937
       int ip;
04938
04939
       /* Extract state vector elements... */
       for (ip = 0; ip < atm->np; ip++)
  if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {</pre>
04940
04941
04942
           value[ip] = gsl_vector_get(x, *n);
04943
04944
04945 }
04946
04948
04949 void y2obs(
04950
       ctl_t * ctl,
       gsl_vector * y,
04951
04952
       obs_t * obs) {
04953
04954
       int id, ir;
04955
       size_t m = 0;
04956
04957
04958
       /\star Decompose measurement vector... \star/
04959
       for (ir = 0; ir < obs->nr; ir++)
04960
        for (id = 0; id < ctl->nd; id++)
04961
           if (gsl_finite(obs->rad[id][ir])) {
04962
            obs->rad[id][ir] = gsl_vector_get(y, m);
04963
             m++;
           }
04964
04965 }
```

# 5.15 jurassic.h File Reference

JURASSIC library declarations.

# **Data Structures**

· struct atm t

Atmospheric data.

struct ctl\_t

Forward model control parameters.

struct los\_t

Line-of-sight data.

• struct obs\_t

Observation geometry and radiance data.

struct tbl\_t

Emissivity look-up tables.

### **Functions**

size\_t atm2x (ctl\_t \*ctl, atm\_t \*atm, gsl\_vector \*x, int \*iqa, int \*ipa)

Compose state vector or parameter vector.

void atm2x\_help (atm\_t \*atm, double zmin, double zmax, double \*value, int val\_iqa, gsl\_vector \*x, int \*iqa, int \*ipa, size t \*n)

Add elements to state vector.

• double brightness (double rad, double nu)

Compute brightness temperature. void cart2geo (double \*x, double \*z, double \*lon, double \*lat) Convert Cartesian coordinates to geolocation. void climatology (ctl t \*ctl, atm t \*atm mean) Interpolate climatological data. double ctmco2 (double nu, double p, double t, double u) Compute carbon dioxide continuum (optical depth). double ctmh2o (double nu, double p, double t, double q, double u) Compute water vapor continuum (optical depth). • double ctmn2 (double nu, double p, double t) Compute nitrogen continuum (absorption coefficient). • double ctmo2 (double nu, double p, double t) Compute oxygen continuum (absorption coefficient). void copy\_atm (ctl\_t \*ctl, atm\_t \*atm\_dest, atm\_t \*atm\_src, int init) Copy and initialize atmospheric data. void copy\_obs (ctl\_t \*ctl, obs\_t \*obs\_dest, obs\_t \*obs\_src, int init) Copy and initialize observation data. int find\_emitter (ctl\_t \*ctl, const char \*emitter) Find index of an emitter. void formod (ctl t \*ctl, atm t \*atm, obs t \*obs) Determine ray paths and compute radiative transfer. void formod\_continua (ctl\_t \*ctl, los\_t \*los, int ip, double \*beta) Compute absorption coefficient of continua. void formod\_fov (ctl\_t \*ctl, obs\_t \*obs) Apply field of view convolution. void formod\_pencil (ctl\_t \*ctl, atm\_t \*atm, obs\_t \*obs, int ir) Compute radiative transfer for a pencil beam. void formod\_srcfunc (ctl\_t \*ctl, tbl\_t \*tbl, double t, double \*src) Compute Planck source function. void geo2cart (double z, double lon, double lat, double \*x) Convert geolocation to Cartesian coordinates. double gravity (double z, double lat) Determine gravity of Earth. void hydrostatic (ctl\_t \*ctl, atm\_t \*atm) Set hydrostatic equilibrium. void idx2name (ctl\_t \*ctl, int idx, char \*quantity) Determine name of state vector quantity for given index. void init\_tbl (ctl\_t \*ctl, tbl\_t \*tbl) Initialize look-up tables. void intpol\_atm (ctl\_t \*ctl, atm\_t \*atm, double z, double \*p, double \*t, double \*q, double \*k) Interpolate atmospheric data. void intpol tbl (ctl t \*ctl, tbl t \*tbl, los t \*los, int ip, double tau path[NG][ND], double tau seg[ND]) Get transmittance from look-up tables. • double intpol\_tbl\_eps (tbl\_t \*tbl, int ig, int id, int ip, int it, double u) Interpolate emissivity from look-up tables. double intpol\_tbl\_u (tbl\_t \*tbl, int ig, int id, int ip, int it, double eps) Interpolate column density from look-up tables. void jsec2time (double jsec, int \*year, int \*mon, int \*day, int \*hour, int \*min, int \*sec, double \*remain) Convert seconds to date.

void kernel (ctl\_t \*ctl, atm\_t \*atm, obs\_t \*obs, gsl\_matrix \*k)

Compute Jacobians.

```
    int locate (double *xx, int n, double x)

          Find array index.

    int locate tbl (float *xx, int n, double x)

          Find array index in float array.

    size_t obs2y (ctl_t *ctl, obs_t *obs, gsl_vector *y, int *ida, int *ira)

          Compose measurement vector.

    double planck (double t, double nu)

          Compute Planck function.

    void raytrace (ctl_t *ctl, atm_t *atm, obs_t *obs, los_t *los, int ir)

          Do ray-tracing to determine LOS.
    • void read_atm (const char *dirname, const char *filename, ctl_t *ctl, atm_t *atm)
          Read atmospheric data.

    void read_ctl (int argc, char *argv[], ctl_t *ctl)

          Read forward model control parameters.

    void read_matrix (const char *dirname, const char *filename, gsl_matrix *matrix)

          Read matrix.

    void read obs (const char *dirname, const char *filename, ctl t *ctl, obs t *obs)

          Read observation data.

    void read_shape (const char *filename, double *x, double *y, int *n)

          Read shape function.
    • double refractivity (double p, double t)
           Compute refractivity (return value is n - 1).
    • double scan_ctl (int argc, char *argv[], const char *varname, int arridx, const char *defvalue, char *value)
          Search control parameter file for variable entry.

    void tangent_point (los_t *los, double *tpz, double *tplon, double *tplat)

           Find tangent point of a given LOS.
    • void time2jsec (int year, int mon, int day, int hour, int min, int sec, double remain, double *jsec)
           Convert date to seconds.

    void timer (const char *name, const char *file, const char *func, int line, int mode)

          Measure wall-clock time.

    void write_atm (const char *dirname, const char *filename, ctl_t *ctl, atm_t *atm)

           Write atmospheric data.

    void write_matrix (const char *dirname, const char *filename, ctl_t *ctl, gsl_matrix *matrix, atm_t *atm,

       obs_t *obs, const char *rowspace, const char *colspace, const char *sort)

    void write_obs (const char *dirname, const char *filename, ctl_t *ctl, obs_t *obs)

           Write observation data.

    void x2atm (ctl t *ctl, gsl vector *x, atm t *atm)

          Decompose parameter vector or state vector.
    • void x2atm_help (atm_t *atm, double zmin, double zmax, double *value, gsl_vector *x, size_t *n)
          Extract elements from state vector.

    void y2obs (ctl_t *ctl, gsl_vector *y, obs_t *obs)

          Decompose measurement vector.
5.15.1 Detailed Description
JURASSIC library declarations.
Definition in file jurassic.h.
```

#### 5.15.2 Function Documentation

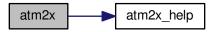
```
5.15.2.1 size_t atm2x ( ctl_t * ctl, atm_t * atm, gsl_vector * x, int * iqa, int * ipa )
```

Compose state vector or parameter vector.

Definition at line 29 of file jurassic.c.

```
00034
00035
00036
        int ig, iw;
00037
00038
        size_t n = 0;
00039
        /* Add pressure... */
00040
00041
        atm2x_help(atm, ctl->retp_zmin, ctl->retp_zmax,
00042
                   atm->p, IDXP, x, iqa, ipa, &n);
00043
       /* Add temperature... */
atm2x_help(atm, ctl->rett_zmin, ctl->rett_zmax,
00044
00045
00046
                   atm->t, IDXT, x, iqa, ipa, &n);
00047
00048
        /* Add volume mixing ratios...
00049
        for (ig = 0; ig < ctl->ng; ig++)
00050
         atm2x_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
00051
                     atm->q[ig], IDXQ(ig), x, iqa, ipa, &n);
00052
00053
        /* Add extinction... */
        for (iw = 0; iw < ctl->nw; iw++)
00055
         atm2x_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
00056
                     atm->k[iw], IDXK(iw), x, iqa, ipa, &n);
00057
00058
        return n;
00059 }
```

Here is the call graph for this function:



5.15.2.2 void atm2x\_help ( atm\_t \* atm, double zmin, double zmax, double \* value, int val\_iqa, gsl\_vector \* x, int \* iqa, int \* ipa, size\_t \* n )

Add elements to state vector.

Definition at line 63 of file jurassic.c.

```
00072
                        {
00073
00074
         int ip;
00075
00076
          /* Add elements to state vector... */
         for (ip = 0; ip < atm->np; ip++)
  if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {
    if (x != NULL)</pre>
00077
00078
00079
08000
                gsl_vector_set(x, *n, value[ip]);
00081
              if (iqa != NULL)
00082
                iqa[*n] = val_iqa;
              if (ipa != NULL)
00083
00084
                ipa[*n] = ip;
00085
              (*n)++;
00086
            }
00087 }
```

## 5.15.2.3 double brightness (double rad, double nu)

Compute brightness temperature.

Definition at line 91 of file jurassic.c.

```
00093 {
00094
00095 return C2 * nu / gsl_log1p(C1 * gsl_pow_3(nu) / rad);
00096 }
```

```
5.15.2.4 void cart2geo ( double * x, double * z, double * lon, double * lat )
```

Convert Cartesian coordinates to geolocation.

Definition at line 101 of file jurassic.c.

### 5.15.2.5 void climatology ( ctl\_t \* ctl, atm\_t \* atm\_mean )

Interpolate climatological data.

Definition at line 117 of file jurassic.c.

```
00119
00120
           static double z[121] = {
00121
             0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,
00122
00123
00124
00125
              56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73,
00126
              74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91,
              92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107,
00127
00128
             108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120
00129
00130
          static double pre[121] = {
  1017, 901.083, 796.45, 702.227, 617.614, 541.644, 473.437, 412.288,
  357.603, 308.96, 265.994, 228.348, 195.619, 167.351, 143.039, 122.198,
  104.369, 89.141, 76.1528, 65.0804, 55.641, 47.591, 40.7233, 34.8637,
00131
00132
00133
00134
              29.8633, 25.5956, 21.9534, 18.8445, 16.1909, 13.9258, 11.9913,
00135
              10.34, 8.92988, 7.72454, 6.6924, 5.80701, 5.04654, 4.39238, 3.82902,
00136
              3.34337, 2.92413, 2.56128, 2.2464, 1.97258, 1.73384, 1.52519, 1.34242, 1.18197, 1.04086, 0.916546, 0.806832, 0.709875, 0.624101, 0.548176,
00137
00138
             0.480974, 0.421507, 0.368904, 0.322408, 0.281386, 0.245249, 0.213465, 0.185549, 0.161072, 0.139644, 0.120913, 0.104568, 0.0903249, 0.0779269,
00139
00140
             0.0671493, 0.0577962, 0.0496902, 0.0426736, 0.0366093, 0.0313743, 0.0268598, 0.0229699, 0.0196206, 0.0167399, 0.0142646, 0.0121397,
00141
00142
              0.0103181, 0.00875775, 0.00742226, 0.00628076, 0.00530519, 0.00447183,
00143
              0.00376124, 0.00315632, 0.00264248, 0.00220738, 0.00184003, 0.00153095, 0.00127204, 0.00105608, 0.000876652, 0.00072798, 0.00060492,
00144
00145
             0.00503201, 0.000419226, 0.000349896, 0.000292659, 0.000245421, 0.000206394, 0.000174125, 0.000147441, 0.000125333, 0.000106985,
00146
00147
              9.173e-05, 7.90172e-05, 6.84172e-05, 5.95574e-05, 5.21183e-05,
00148
              4.58348e-05, 4.05127e-05, 3.59987e-05, 3.21583e-05, 2.88718e-05,
00149
00150
              2.60322e-05, 2.35687e-05, 2.14263e-05, 1.95489e-05
00151
00152
00153
           static double tem[121] = {
00154
             285.14, 279.34, 273.91, 268.3, 263.24, 256.55, 250.2, 242.82, 236.17,
00155
              229.87, 225.04, 221.19, 218.85, 217.19, 216.2, 215.68, 215.42, 215.55,
```

```
215.92, 216.4, 216.93, 217.45, 218, 218.68, 219.39, 220.25, 221.3,
                     222.41, 223.88, 225.42, 227.2, 229.52, 231.89, 234.51, 236.85, 239.42, 241.94, 244.57, 247.36, 250.32, 253.34, 255.82, 258.27, 260.39,
00157
00158
                     262.03, 263.45, 264.2, 264.78, 264.67, 264.38, 263.24, 262.03, 260.02,
00159
00160
                     258.09, 255.63, 253.28, 250.43, 247.81, 245.26, 242.77, 240.38, 237.94, 235.79, 233.53, 231.5, 229.53, 227.6, 225.62, 223.77, 222.06,
00161
                     220.33, 218.69, 217.18, 215.64, 214.13, 212.52, 210.86, 209.25,
00162
                     207.49, 205.81, 204.11, 202.22, 200.32, 198.39, 195.92, 193.46,
00163
                    190.94, 188.31, 185.82, 183.57, 181.43, 179.74, 178.64, 178.1, 178.25, 178.7, 179.41, 180.67, 182.31, 184.18, 186.6, 189.53, 192.66, 196.54, 201.13, 205.93, 211.73, 217.86, 225, 233.53, 242.57, 252.14, 261.48, 272.97, 285.26, 299.12, 312.2, 324.17, 338.34, 352.56, 365.28
00164
00165
00166
00167
00168
00169
00170
                 static double c2h2[121] = {
                    1.352e-09, 2.83e-10, 1.269e-10, 6.926e-11, 4.346e-11, 2.909e-11,
00171
                     2.014e-11, 1.363e-11, 8.71e-12, 5.237e-12, 2.718e-12, 1.375e-12, 5.786e-13, 2.16e-13, 7.317e-14, 2.551e-14, 1.055e-14, 4.758e-15,
00172
00173
                     2.056e-15, 7.703e-16, 2.82e-16, 1.035e-16, 4.382e-17, 1.946e-17,
                     9.638e-18, 5.2e-18, 2.811e-18, 1.494e-18, 7.925e-19, 4.213e-19,
00175
00176
                     1.998e-19, 8.78e-20, 3.877e-20, 1.728e-20, 7.743e-21, 3.536e-21
00177
                     1.623e-21, 7.508e-22, 3.508e-22, 1.65e-22, 7.837e-23, 3.733e-23,
00178
                     1.808e-23, 8.77e-24, 4.285e-24, 2.095e-24, 1.032e-24, 5.082e-25,
                     00179
00180
                     00182
00183
00184
00185
                static double c2h6[121] = {
00186
                   2.667e-09, 2.02e-09, 1.658e-09, 1.404e-09, 1.234e-09, 1.109e-09,
00187
                     1.012e-09, 9.262e-10, 8.472e-10, 7.71e-10, 6.932e-10, 6.216e-10,
                     5.503e-10, 4.87e-10, 4.342e-10, 3.861e-10, 3.347e-10, 2.772e-10,
00188
00189
                     2.209e-10, 1.672e-10, 1.197e-10, 8.536e-11, 5.783e-11, 3.846e-11,
                     2.495e-11, 1.592e-11, 1.017e-11, 6.327e-12, 3.895e-12, 2.403e-12, 1.416e-12, 8.101e-13, 4.649e-13, 2.686e-13, 1.557e-13, 9.14e-14,
00190
00191
                     5.386e-14, 3.19e-14, 1.903e-14, 1.14e-14, 6.875e-15, 4.154e-15, 2.538e-15, 1.553e-15, 9.548e-16, 5.872e-16, 3.63e-16, 2.244e-16,
00192
00194
                     1.388e-16, 8.587e-17, 5.308e-17, 3.279e-17, 2.017e-17, 1.238e-17,
00195
                     7.542e-18, 4.585e-18, 2.776e-18, 1.671e-18, 9.985e-19, 5.937e-19,
00196
                     3.518e-19, 2.07e-19, 1.215e-19, 7.06e-20, 4.097e-20, 2.37e-20,
                     1.363e-20, 7.802e-21, 4.441e-21, 2.523e-21, 1.424e-21, 8.015e-22,
00197
                     4.497e-22, 2.505e-22, 1.391e-22, 7.691e-23, 4.238e-23, 2.331e-23,
00198
                     1.274e-23, 6.929e-24, 3.752e-24, 2.02e-24, 1.083e-24, 5.774e-25,
00199
                     3.041e-25, 1.593e-25, 8.308e-26, 4.299e-26, 2.195e-26, 1.112e-26,
00200
00201
                     00202
                    0, 0, 0, 0, 0, 0, 0, 0
00203
                };
00204
00205
                static double cc14[121] = {
                   1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10, 1.075e-10,
                     1.075e-10, 1.075e-10, 1.075e-10, 1.06e-10, 1.024e-10, 9.69e-11,
00207
00208
                     8.93e-11, 8.078e-11, 7.213e-11, 6.307e-11, 5.383e-11, 4.49e-11,
                    3.609e-11, 2.705e-11, 1.935e-11, 1.385e-11, 8.35e-12, 5.485e-12, 3.853e-12, 2.22e-12, 5.875e-13, 3.445e-13, 1.015e-13, 6.075e-14, 4.383e-14, 2.692e-14, 1e-14, 1e-1
00209
00210
00211
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00213
00214
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00215
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00216
                     1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
                     le-14, le
00217
                      1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14, 1e-14,
00219
00220
                     1e-14, 1e-14, 1e-14
00221
00222
00223
                static double ch4[121] = {
                   1.864e-06, 1.835e-06, 1.819e-06, 1.805e-06, 1.796e-06, 1.788e-06,
00225
                     1.782e-06, 1.776e-06, 1.769e-06, 1.761e-06, 1.749e-06, 1.734e-06,
                     1.716e-06, 1.692e-06, 1.654e-06, 1.61e-06, 1.567e-06, 1.502e-06,
00226
00227
                     1.433e-06, 1.371e-06, 1.323e-06, 1.277e-06, 1.232e-06, 1.188e-06,
                     1.147e-06, 1.108e-06, 1.07e-06, 1.027e-06, 9.854e-07, 9.416e-07, 8.933e-07, 8.478e-07, 7.988e-07, 7.515e-07, 7.07e-07, 6.64e-07, 6.239e-07, 5.864e-07, 5.512e-07, 5.184e-07, 4.87e-07, 4.571e-07,
00228
00229
00230
                     4.296e-07, 4.04e-07, 3.802e-07, 3.578e-07, 3.383e-07, 3.203e-07, 3.032e-07, 2.889e-07, 2.76e-07, 2.635e-07, 2.519e-07, 2.409e-07,
00231
00232
                     2.302e-07, 2.219e-07, 2.144e-07, 2.071e-07, 1.999e-07, 1.93e-07, 1.862e-07, 1.795e-07, 1.731e-07, 1.668e-07, 1.607e-07, 1.548e-07,
00233
00234
                     1.49e-07, 1.434e-07, 1.38e-07, 1.328e-07, 1.277e-07, 1.227e-07, 1.18e-07, 1.134e-07, 1.089e-07, 1.046e-07, 1.004e-07, 9.635e-08,
00235
00236
                      9.245e-08, 8.867e-08, 8.502e-08, 8.15e-08, 7.809e-08, 7.48e-08,
                     7.159e-08, 6.849e-08, 6.55e-08, 6.262e-08, 5.98e-08, 5.708e-08,
00238
00239
                     5.448e-08, 5.194e-08, 4.951e-08, 4.72e-08, 4.5e-08, 4.291e-08,
00240
                     4.093e-08, 3.905e-08, 3.729e-08, 3.563e-08, 3.408e-08, 3.265e-08,
                     3.128e-08, 2.996e-08, 2.87e-08, 2.76e-08, 2.657e-08, 2.558e-08, 2.467e-08, 2.385e-08, 2.307e-08, 2.234e-08, 2.168e-08, 2.108e-08,
00241
00242
```

```
2.05e-08, 1.998e-08, 1.947e-08, 1.902e-08, 1.86e-08, 1.819e-08,
00244
00245
00246
00247
             static double clo[121] = {
                 7.419e-15, 1.061e-14, 1.518e-14, 2.195e-14, 3.175e-14, 4.666e-14,
00248
                 6.872e-14, 1.03e-13, 1.553e-13, 2.375e-13, 3.664e-13, 5.684e-13,
                 8.915e-13, 1.402e-12, 2.269e-12, 4.125e-12, 7.501e-12, 1.257e-11,
00250
00251
                2.048e-11, 3.338e-11, 5.44e-11, 8.846e-11, 1.008e-10, 1.082e-10,
00252
                 1.157e-10, 1.232e-10, 1.312e-10, 1.539e-10, 1.822e-10, 2.118e-10,
                 2.387e-10, 2.687e-10, 2.875e-10, 3.031e-10, 3.23e-10, 3.648e-10,
00253
00254
                 4.117e-10, 4.477e-10, 4.633e-10, 4.794e-10, 4.95e-10, 5.104e-10,
00255
                 5.259e-10, 5.062e-10, 4.742e-10, 4.443e-10, 4.051e-10, 3.659e-10,
                 3.305e-10, 2.911e-10, 2.54e-10, 2.215e-10, 1.927e-10, 1.675e-10,
00256
00257
                 1.452e-10, 1.259e-10, 1.09e-10, 9.416e-11, 8.119e-11, 6.991e-11,
00258
                  6.015 e^{-11}, \ 5.163 e^{-11}, \ 4.43 e^{-11}, \ 3.789 e^{-11}, \ 3.24 e^{-11}, \ 2.769 e^{-11}, \\
00259
                2.361e-11, 2.011e-11, 1.71e-11, 1.453e-11, 1.233e-11, 1.045e-11,
                8.851e-12, 7.48e-12, 6.316e-12, 5.326e-12, 4.487e-12, 3.778e-12, 3.176e-12, 2.665e-12, 2.234e-12, 1.87e-12, 1.563e-12, 1.304e-12,
00260
00262
                 1.085e-12, 9.007e-13, 7.468e-13, 6.179e-13, 5.092e-13, 4.188e-13,
                 3.442e-13, 2.816e-13, 2.304e-13, 1.885e-13, 1.542e-13,
00263
                 1.035e-13, 8.5e-14, 7.004e-14, 5.783e-14, 4.795e-14, 4.007e-14,
00264
                3.345e-14, 2.792e-14, 2.33e-14, 1.978e-14, 1.686e-14, 1.438e-14, 1.234e-14, 1.07e-14, 9.312e-15, 8.131e-15, 7.164e-15, 6.367e-15, 5.67e-15, 5.088e-15, 4.565e-15, 4.138e-15, 3.769e-15, 3.432e-15,
00265
00266
00267
00268
                 3.148e-15
00269
00270
00271
             static double clono2[121] = {
                1.011e-13, 1.515e-13, 2.272e-13, 3.446e-13, 5.231e-13, 8.085e-13, 1.253e-12, 1.979e-12, 3.149e-12, 5.092e-12, 8.312e-12, 1.366e-11, 2.272e-11, 3.791e-11, 6.209e-11, 9.101e-11, 1.334e-10, 1.951e-10,
00272
00273
00274
00275
                 2.853e-10, 3.94e-10, 4.771e-10, 5.771e-10, 6.675e-10, 7.665e-10,
00276
                 8.504e-10, 8.924e-10, 9.363e-10, 8.923e-10, 8.411e-10, 7.646e-10,
00277
                 6.525e-10, 5.576e-10, 4.398e-10, 3.403e-10, 2.612e-10, 1.915e-10,
                1.407e-10, 1.028e-10, 7.455e-11, 5.42e-11, 3.708e-11, 2.438e-11, 1.618e-11, 1.075e-11, 7.17e-12, 4.784e-12, 3.205e-12, 2.147e-12, 1.44e-12, 9.654e-13, 6.469e-13, 4.332e-13, 2.891e-13, 1.926e-13,
00278
00279
00281
                 1.274e-13, 8.422e-14, 5.547e-14, 3.636e-14, 2.368e-14, 1.536e-14,
                 9.937e-15, 6.39e-15, 4.101e-15, 2.61e-15, 1.659e-15, 1.052e-15,
00282
00283
                 6.638e-16, 4.172e-16, 2.61e-16, 1.63e-16, 1.013e-16, 6.275e-17,
                 3.879e-17, 2.383e-17, 1.461e-17, 8.918e-18, 5.43e-18, 3.301e-18,
00284
                 1.997e-18, 1.203e-18, 7.216e-19, 4.311e-19, 2.564e-19, 1.519e-19,
00285
00286
                 8.911e-20, 5.203e-20, 3.026e-20, 1.748e-20, 9.99e-21, 5.673e-21,
                 3.215e-21, 1.799e-21, 1.006e-21, 5.628e-22, 3.146e-22, 1.766e-22, 9.94e-23, 5.614e-23, 3.206e-23, 1.841e-23, 1.071e-23, 6.366e-24,
00288
00289
                 3.776e-24, 2.238e-24, 1.326e-24, 8.253e-25, 5.201e-25, 3.279e-25,
                2.108e-25, 1.395e-25, 9.326e-26, 6.299e-26, 4.365e-26, 3.104e-26, 2.219e-26, 1.621e-26, 1.185e-26, 8.92e-27, 6.804e-27, 5.191e-27,
00290
00291
00292
                 4.041e-27
00293
00294
00295
             static double co[121] = {
               1.907e-07, 1.553e-07, 1.362e-07, 1.216e-07, 1.114e-07, 1.036e-07, 9.737e-08, 9.152e-08, 8.559e-08, 7.966e-08, 7.277e-08, 6.615e-08,
00296
00297
                 5.884e-08, 5.22e-08, 4.699e-08, 4.284e-08, 3.776e-08, 3.274e-08,
00298
                 2.845e-08, 2.479e-08, 2.246e-08, 2.054e-08, 1.991e-08, 1.951e-08,
                 1.94e-08, 2.009e-08, 2.1e-08, 2.201e-08, 2.322e-08, 2.45e-08,
00300
00301
                 2.602e-08, 2.73e-08, 2.867e-08, 2.998e-08, 3.135e-08, 3.255e-08,
00302
                 3.352e-08, 3.426e-08, 3.484e-08, 3.53e-08, 3.593e-08, 3.671e-08,
00303
                 3.759e-08, 3.945e-08, 4.192e-08, 4.49e-08, 5.03e-08, 5.703e-08,
                 6.538e-08, 7.878e-08, 9.644e-08, 1.196e-07, 1.498e-07, 1.904e-07,
00304
00305
                 2.422e-07, 3.055e-07, 3.804e-07, 4.747e-07, 5.899e-07, 7.272e-07,
                8.91e-07, 1.071e-06, 1.296e-06, 1.546e-06, 1.823e-06, 2.135e-06, 2.44e-06, 2.714e-06, 2.967e-06, 3.189e-06, 3.391e-06, 3.58e-06,
00306
00307
00308
                 3.773e-06, 4.022e-06, 4.346e-06, 4.749e-06, 5.199e-06, 5.668e-06,
00309
                 6.157e-06, 6.688e-06, 7.254e-06, 7.867e-06, 8.539e-06, 9.26e-06,
                 1.009e-05, 1.119e-05, 1.228e-05, 1.365e-05, 1.506e-05, 1.641e-05,
00310
00311
                 1.784e-05, 1.952e-05, 2.132e-05, 2.323e-05, 2.531e-05, 2.754e-05,
                 3.047e-05, 3.459e-05, 3.922e-05, 4.439e-05, 4.825e-05, 5.077e-05,
                 5.34e-05, 5.618e-05, 5.909e-05, 6.207e-05, 6.519e-05, 6.845e-05,
00313
                 6.819e-05, 6.726e-05, 6.622e-05, 6.512e-05, 6.671e-05, 6.862e-05, 7.048e-05, 7.264e-05, 7.3e-05, 7.2e-05, 7.2e-
00314
00315
00316
00317
00318
             static double cof2[121] = {
                 7.5e-14, 1.055e-13, 1.485e-13, 2.111e-13, 3.001e-13, 4.333e-13,
00319
                 6.269e-13, 9.221e-13, 1.364e-12, 2.046e-12, 3.093e-12, 4.703e-12, 7.225e-12, 1.113e-11, 1.66e-11, 2.088e-11, 2.626e-11, 3.433e-11,
00320
00321
                 4.549e-11, 5.886e-11, 7.21e-11, 8.824e-11, 1.015e-10, 1.155e-10, 1.288e-10, 1.388e-10, 1.497e-10, 1.554e-10, 1.606e-10, 1.639e-10,
00322
00323
                 1.64e-10, 1.64e-10, 1.596e-10, 1.542e-10, 1.482e-10, 1.382e-10,
                 1.289e-10, 1.198e-10, 1.109e-10, 1.026e-10, 9.484e-11, 8.75e-11, 8.086e-11, 7.49e-11, 6.948e-11, 6.446e-11, 5.961e-11, 5.505e-11,
00325
00326
00327
                 5.085e-11, 4.586e-11, 4.1e-11, 3.665e-11, 3.235e-11, 2.842e-11,
                 2.491e-11, 2.11e-11, 1.769e-11, 1.479e-11, 1.197e-11, 9.631e-12, 7.74e-12, 6.201e-12, 4.963e-12, 3.956e-12, 3.151e-12, 2.507e-12,
00328
00329
```

```
1.99e-12, 1.576e-12, 1.245e-12, 9.83e-13, 7.742e-13, 6.088e-13,
                        4.782e-13, 3.745e-13, 2.929e-13, 2.286e-13, 1.782e-13, 1.388e-13, 1.079e-13, 8.362e-14, 6.471e-14, 4.996e-14, 3.85e-14, 2.96e-14,
00331
00332
00333
                         2.265e-14, 1.729e-14, 1.317e-14, 9.998e-15, 7.549e-15, 5.683e-15,
                         4.273e-15, 3.193e-15, 2.385e-15, 1.782e-15, 1.331e-15, 9.957e-16,
00334
                         7.461e-16, 5.601e-16, 4.228e-16, 3.201e-16, 2.438e-16, 1.878e-16,
00335
                         1.445e-16, 1.111e-16, 8.544e-17, 6.734e-17, 5.341e-17, 4.237e-17,
                         3.394e-17, 2.759e-17, 2.254e-17, 1.851e-17, 1.54e-17, 1.297e-17,
00337
00338
                         1.096e-17, 9.365e-18, 8e-18, 6.938e-18, 6.056e-18, 5.287e-18,
00339
                         4.662e-18
00340
00341
00342
                   static double f11[121] = {
                      2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10,
00343
00344
                         2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.65e-10, 2.635e-10, 2.536e-10,
00345
                         2.44e-10, 2.348e-10, 2.258e-10, 2.153e-10, 2.046e-10, 1.929e-10,
                         1.782e-10, 1.648e-10, 1.463e-10, 1.291e-10, 1.1e-10, 8.874e-11,
00346
                         7.165e-11, 5.201e-11, 3.744e-11, 2.577e-11, 1.64e-11, 1.048e-11,
00347
                         5.993e-12, 3.345e-12, 1.839e-12, 9.264e-13, 4.688e-13, 2.329e-13,
00349
                         1.129e-13, 5.505e-14, 2.825e-14, 1.492e-14, 7.997e-15, 5.384e-15,
00350
                         3.988e-15, 2.955e-15, 2.196e-15, 1.632e-15, 1.214e-15, 9.025e-16,
00351
                         6.708e-16, 4.984e-16, 3.693e-16, 2.733e-16, 2.013e-16, 1.481e-16,
                        1.087e-16, 7.945e-17, 5.782e-17, 4.195e-17, 3.038e-17, 2.19e-17, 1.577e-17, 1.128e-17, 8.063e-18, 5.753e-18, 4.09e-18, 2.899e-18, 2.048e-18, 1.444e-18, 1.015e-18, 7.12e-19, 4.985e-19, 3.474e-19,
00352
00353
00354
                         2.417e-19, 1.677e-19, 1.161e-19, 8.029e-20, 5.533e-20, 3.799e-20,
00355
                         2.602e-20, 1.776e-20, 1.209e-20, 8.202e-21, 5.522e-21, 3.707e-21,
00356
00357
                         2.48e-21, 1.652e-21, 1.091e-21, 7.174e-22, 4.709e-22, 3.063e-22,
00358
                         1.991e-22, 1.294e-22, 8.412e-23, 5.483e-23, 3.581e-23, 2.345e-23,
00359
                         1.548e-23, 1.027e-23, 6.869e-24, 4.673e-24, 3.173e-24, 2.153e-24,
00360
                         1.461e-24, 1.028e-24, 7.302e-25, 5.188e-25, 3.739e-25, 2.753e-25,
00361
                         2.043e-25, 1.528e-25, 1.164e-25, 9.041e-26, 7.051e-26, 5.587e-26,
                         4.428e-26, 3.588e-26, 2.936e-26, 2.402e-26, 1.995e-26
00362
00363
00364
                   static double f12[121] = {
00365
                        5.45e-10, 5.45e-
00366
00368
                         5.155e-10, 5.022e-10, 4.893e-10, 4.772e-10, 4.655e-10, 4.497e-10,
00369
                         4.249e-10, 4.015e-10, 3.632e-10, 3.261e-10, 2.858e-10,
                                                                                                                                                                  2.408e-10
00370
                         2.03e-10, 1.685e-10, 1.4e-10, 1.163e-10, 9.65e-11, 8.02e-11, 6.705e-11,
                        5.624e-11, 4.764e-11, 4.249e-11, 3.792e-11, 3.315e-11, 2.819e-11, 2.4e-11, 1.999e-11, 1.64e-11, 1.352e-11, 1.14e-11, 9.714e-12,
00371
00372
                         8.28e-12, 7.176e-12, 6.251e-12, 5.446e-12, 4.72e-12, 4.081e-12,
00373
                         3.528e-12, 3.08e-12, 2.699e-12, 2.359e-12, 2.111e-12, 1.901e-12,
00374
00375
                         1.709e-12, 1.534e-12, 1.376e-12, 1.233e-12, 1.103e-12, 9.869e-13,
00376
                         8.808e-13, 7.859e-13, 7.008e-13, 6.241e-13, 5.553e-13, 4.935e-13,
                         4.383e-13, 3.889e-13, 3.447e-13, 3.054e-13, 2.702e-13, 2.389e-13, 2.11e-13, 1.862e-13, 1.643e-13, 1.448e-13, 1.274e-13, 1.121e-13, 9.844e-14, 8.638e-14, 7.572e-14, 6.62e-14, 5.782e-14, 5.045e-14,
00377
00378
00379
                         4.394e-14, 3.817e-14, 3.311e-14, 2.87e-14, 2.48e-14, 2.142e-14,
                         1.851e-14, 1.599e-14, 1.383e-14, 1.196e-14, 1.036e-14, 9e-15,
00381
00382
                         7.828e-15, 6.829e-15, 5.992e-15, 5.254e-15, 4.606e-15, 4.037e-15,
                        3.583e-15, 3.19e-15, 2.841e-15, 2.542e-15, 2.291e-15, 2.07e-15, 1.875e-15, 1.71e-15, 1.57e-15, 1.442e-15, 1.333e-15, 1.232e-15,
00383
00384
00385
                         1.147e-15, 1.071e-15, 1.001e-15, 9.396e-16
00387
00388
                   static double f14[121] = {
                        9e-11, 8.73e-11, 8.46e-11, 8.19e-11, 7.92e-11, 7.74e-11, 7.65e-11, 7
00389
00390
00391
00392
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-
00393
00394
00395
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                    7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                                                                                                 7.65e-11,
00396
                         7.65e-11, 7.65e-11, 7.65e-11,
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00397
                                                                                                                                                                                 7.65e-11,
00398
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
00399
00400
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                                                                                                 7.65e-11,
                                                                                                                                                                                7.65e-11,
00401
                         7.65e-11, 7.65e-11, 7.65e-11,
                                                                                                    7.65e-11, 7.65e-11, 7.65e-11,
00402
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11,
                         7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11, 7.65e-11
00403
00404
00405
00406
00407
00408
                    static double f22[121] = {
                        1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10, 1.4e-10,
00409
                         1.4e-10, 1.4e-10, 1.4e-10, 1.372e-10, 1.317e-10, 1.235e-10, 1.153e-10,
00410
                         1.075e-10, 1.002e-10, 9.332e-11, 8.738e-11, 8.194e-11, 7.7e-11,
                         7.165e-11, 6.753e-11, 6.341e-11, 5.971e-11, 5.6e-11, 5.229e-11, 4.859e-11, 4.488e-11, 4.118e-11, 3.83e-11, 3.568e-11, 3.308e-11,
00412
00413
00414
                         3.047e-11,\ 2.82e-11,\ 2.594e-11,\ 2.409e-11,\ 2.237e-11,\ 2.065e-11,
                         1.894e-11, 1.771e-11, 1.647e-11, 1.532e-11, 1.416e-11, 1.332e-11, 1.246e-11, 1.161e-11, 1.087e-11, 1.017e-11, 9.471e-12, 8.853e-12,
00415
00416
```

```
8.235e-12, 7.741e-12, 7.247e-12, 6.836e-12, 6.506e-12, 6.176e-12,
            5.913e-12, 5.65e-12, 5.419e-12, 5.221e-12, 5.024e-12, 4.859e-12,
00418
00419
            4.694e-12, 4.546e-12, 4.414e-12, 4.282e-12, 4.15e-12, 4.019e-12,
            3.903e-12, 3.805e-12, 3.706e-12, 3.607e-12, 3.508e-12, 3.41e-12, 3.31e-12, 3.212e-12, 3.129e-12, 3.047e-12, 2.964e-12, 2.882e-12, 2.8e-12, 2.734e-12, 2.668e-12, 2.602e-12, 2.537e-12, 2.471e-12,
00420
00421
00422
            2.421e-12, 2.372e-12, 2.322e-12, 2.273e-12, 2.224e-12, 2.182e-12,
            2.141e-12, 2.1e-12, 2.059e-12, 2.018e-12, 1.977e-12, 1.935e-12,
00424
00425
            1.894e-12, 1.853e-12, 1.812e-12, 1.77e-12, 1.73e-12, 1.688e-12,
00426
            1.647e-12, 1.606e-12, 1.565e-12, 1.524e-12, 1.483e-12, 1.441e-12,
           1.4e-12, 1.359e-12, 1.317e-12, 1.276e-12, 1.235e-12, 1.194e-12, 1.153e-12, 1.112e-12, 1.071e-12, 1.029e-12, 9.883e-13
00427
00428
00429
00430
00431
         static double h2o[121] = {
           0.01166, 0.008269, 0.005742, 0.003845, 0.00277, 0.001897, 0.001272, 0.000827, 0.000539, 0.0003469, 0.0001579, 3.134e-05, 1.341e-05,
00432
00433
            6.764e-06, 4.498e-06, 3.703e-06, 3.724e-06, 3.899e-06, 4.002e-06,
00434
            4.122e-06, 4.277e-06, 4.438e-06, 4.558e-06, 4.673e-06, 4.763e-06,
00436
            4.809e-06, 4.856e-06, 4.936e-06, 5.021e-06, 5.114e-06, 5.222e-06,
            5.331e-06, 5.414e-06, 5.488e-06, 5.563e-06, 5.633e-06, 5.704e-06,
00437
00438
            5.767e-06, 5.819e-06, 5.872e-06, 5.914e-06, 5.949e-06, 5.984e-06,
00439
            6.015e-06, 6.044e-06, 6.073e-06, 6.104e-06, 6.136e-06, 6.167e-06,
           6.189e-06, 6.208e-06, 6.226e-06, 6.212e-06, 6.185e-06, 6.158e-06, 6.114e-06, 6.066e-06, 6.018e-06, 5.877e-06, 5.728e-06, 5.582e-06,
00440
00441
            5.437e-06, 5.296e-06, 5.156e-06, 5.02e-06, 4.886e-06, 4.754e-06,
            4.625e-06, 4.498e-06, 4.374e-06, 4.242e-06, 4.096e-06, 3.955e-06,
00443
00444
            3.817e-06, 3.683e-06, 3.491e-06, 3.204e-06, 2.94e-06, 2.696e-06,
           2.47e-06, 2.252e-06, 2.019e-06, 1.808e-06, 1.618e-06, 1.445e-06, 1.285e-06, 1.105e-06, 9.489e-07, 8.121e-07, 6.938e-07, 5.924e-07,
00445
00446
            5.04e-07, 4.288e-07, 3.648e-07, 3.103e-07, 2.642e-07, 2.252e-07,
00447
00448
            1.921e-07, 1.643e-07, 1.408e-07, 1.211e-07, 1.048e-07, 9.063e-08,
            7.835e-08, 6.774e-08, 5.936e-08, 5.221e-08, 4.592e-08, 4.061e-08,
00449
00450
            3.62e-08, 3.236e-08, 2.902e-08, 2.62e-08, 2.383e-08, 2.171e-08,
00451
           1.989e-08, 1.823e-08, 1.684e-08, 1.562e-08, 1.449e-08, 1.351e-08
00452
00453
         static double h2o2[121] =
00455
           1.779e-10, 7.938e-10, 8.953e-10, 8.032e-10, 6.564e-10, 5.159e-10,
            4.003e-10, 3.026e-10, 2.222e-10, 1.58e-10, 1.044e-10, 6.605e-11,
00456
00457
           3.413e-11, 1.453e-11, 1.062e-11, 1.009e-11, 9.597e-12, 1.175e-11,
           1.572e-11, 2.091e-11, 2.746e-11, 3.603e-11, 4.791e-11, 6.387e-11, 8.239e-11, 1.007e-10, 1.23e-10, 1.363e-10, 1.489e-10, 1.585e-10, 1.608e-10, 1.632e-10, 1.576e-10, 1.502e-10, 1.423e-10, 1.302e-10,
00458
00459
00460
            1.192e-10, 1.085e-10, 9.795e-11, 8.854e-11, 8.057e-11, 7.36e-11, 6.736e-11, 6.362e-11, 6.087e-11, 5.825e-11, 5.623e-11, 5.443e-11,
00462
00463
           5.27e-11, 5.098e-11, 4.931e-11, 4.769e-11, 4.611e-11, 4.458e-11,
            4.308e-11, 4.102e-11, 3.887e-11, 3.682e-11, 3.521e-11, 3.369e-11, 3.224e-11, 3.082e-11, 2.946e-11, 2.814e-11, 2.687e-11, 2.566e-11,
00464
00465
            2.449e-11, 2.336e-11, 2.227e-11, 2.123e-11, 2.023e-11, 1.927e-11,
00466
            1.835e-11, 1.746e-11, 1.661e-11, 1.58e-11, 1.502e-11, 1.428e-11,
            1.357e-11, 1.289e-11, 1.224e-11, 1.161e-11, 1.102e-11, 1.045e-11,
00468
00469
            9.895e-12, 9.369e-12, 8.866e-12, 8.386e-12, 7.922e-12, 7.479e-12,
            7.06e-12, 6.656e-12, 6.274e-12, 5.914e-12, 5.575e-12, 5.257e-12, 4.959e-12, 4.679e-12, 4.42e-12, 4.178e-12, 3.954e-12, 3.75e-12,
00470
00471
            3.557e-12, 3.372e-12, 3.198e-12, 3.047e-12, 2.908e-12, 2.775e-12, 2.653e-12, 2.544e-12, 2.442e-12, 2.346e-12, 2.26e-12, 2.183e-12,
00472
            2.11e-12, 2.044e-12, 1.98e-12, 1.924e-12, 1.871e-12, 1.821e-12,
00474
00475
           1.775e-12
00476
00477
00478
         static double hcn[121] = {
00479
           5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10,
            5.5e-10, 5.5e-10, 5.5e-10, 5.5e-10, 5.498e-10, 5.495e-10, 5.493e-10,
00480
00481
            5.49e-10, 5.488e-10, 4.717e-10, 3.946e-10, 3.174e-10, 2.4e-10,
00482
           1.626e-10, 1.619e-10, 1.612e-10, 1.602e-10, 1.593e-10, 1.582e-10,
00483
            1.572e-10, 1.56e-10, 1.549e-10, 1.539e-10, 1.53e-10, 1.519e-10,
            1.506e-10, 1.487e-10, 1.467e-10, 1.449e-10, 1.43e-10, 1.413e-10,
00484
            1.397e-10, 1.382e-10, 1.368e-10, 1.354e-10, 1.337e-10, 1.315e-10,
00485
            1.292e-10, 1.267e-10, 1.241e-10, 1.215e-10, 1.19e-10, 1.165e-10,
            1.141e-10, 1.118e-10, 1.096e-10, 1.072e-10, 1.047e-10, 1.021e-10,
00487
00488
            9.968e-11, 9.739e-11, 9.539e-11, 9.339e-11, 9.135e-11, 8.898e-11,
00489
            8.664e-11, 8.439e-11, 8.249e-11, 8.075e-11, 7.904e-11, 7.735e-11,
            7.565e-11, 7.399e-11, 7.245e-11, 7.109e-11, 6.982e-11, 6.863e-11,
00490
            6.755e-11, 6.657e-11, 6.587e-11, 6.527e-11, 6.476e-11, 6.428e-11,
00491
            6.382e-11, 6.343e-11, 6.307e-11, 6.272e-11, 6.238e-11, 6.205e-11,
00492
            6.17e-11, 6.137e-11, 6.102e-11, 6.072e-11, 6.046e-11, 6.03e-11,
00493
00494
            6.018e-11, 6.01e-11, 6.001e-11, 5.992e-11, 5.984e-11, 5.975e-11,
00495
            5.967e-11, 5.958e-11, 5.95e-11, 5.941e-11, 5.933e-11, 5.925e-11,
           5.916e-11, 5.908e-11, 5.899e-11, 5.891e-11, 5.883e-11, 5.874e-11, 5.866e-11, 5.858e-11, 5.85e-11, 5.841e-11, 5.833e-11, 5.825e-11, 5.817e-11, 5.808e-11, 5.8e-11, 5.792e-11, 5.784e-11
00496
00497
00498
00499
00500
00501
         static double hno3[121] = {
           1.809e-10, 7.234e-10, 5.899e-10, 4.342e-10, 3.277e-10, 2.661e-10, 2.35e-10, 2.267e-10, 2.389e-10, 2.651e-10, 3.255e-10, 4.099e-10,
00502
00503
```

```
5.42e-10, 6.978e-10, 8.807e-10, 1.112e-09, 1.405e-09, 2.04e-09,
                3.111e-09, 4.5e-09, 5.762e-09, 7.37e-09, 7.852e-09, 8.109e-09,
00505
                8.067e-09, 7.554e-09, 7.076e-09, 6.268e-09, 5.524e-09, 4.749e-09,
00506
00507
                3.909e-09, 3.223e-09, 2.517e-09, 1.942e-09, 1.493e-09, 1.122e-09,
                8.449e-10, 6.361e-10, 4.787e-10, 3.611e-10, 2.804e-10, 2.215e-10,
00508
00509
                1.758e-10, 1.441e-10, 1.197e-10, 9.953e-11, 8.505e-11, 7.334e-11,
                6.325e-11, 5.625e-11, 5.058e-11, 4.548e-11, 4.122e-11, 3.748e-11,
                3.402e-11, 3.088e-11, 2.8e-11, 2.536e-11, 2.293e-11, 2.072e-11,
00511
00512
                1.871e-11, 1.687e-11, 1.52e-11, 1.368e-11, 1.23e-11, 1.105e-11,
00513
                9.922e-12, 8.898e-12, 7.972e-12, 7.139e-12, 6.385e-12, 5.708e-12,
                5.099e-12, 4.549e-12, 4.056e-12, 3.613e-12, 3.216e-12, 2.862e-12,
00514
00515
                2.544e-12, 2.259e-12, 2.004e-12, 1.776e-12, 1.572e-12, 1.391e-12,
00516
                1.227e-12, 1.082e-12, 9.528e-13, 8.379e-13, 7.349e-13, 6.436e-13,
                5.634e-13, 4.917e-13, 4.291e-13, 3.745e-13, 3.267e-13, 2.854e-13,
00517
00518
                2.494e-13, 2.181e-13, 1.913e-13, 1.68e-13, 1.479e-13, 1.31e-13,
                1.159e-13, 1.025e-13, 9.067e-14, 8.113e-14, 7.281e-14, 6.535e-14, 5.892e-14, 5.348e-14, 4.867e-14, 4.439e-14, 4.073e-14, 3.76e-14,
00519
00520
                3.476e-14, 3.229e-14, 3e-14, 2.807e-14, 2.635e-14, 2.473e-14,
00521
                2.332e-14
00523
00524
00525
             static double hno4[121] = {
               6.118e-12, 3.594e-12, 2.807e-12, 3.04e-12, 4.458e-12, 7.986e-12,
00526
                1.509e-11, 2.661e-11, 3.738e-11, 4.652e-11, 4.429e-11, 3.992e-11, 3.347e-11, 3.005e-11, 3.173e-11, 4.055e-11, 5.812e-11, 8.489e-11,
00527
00528
                1.19e-10, 1.482e-10, 1.766e-10, 2.103e-10, 2.35e-10, 2.598e-10,
00529
00530
                2.801e-10, 2.899e-10, 3e-10, 2.817e-10, 2.617e-10, 2.332e-10,
00531
                1.933e-10, 1.605e-10, 1.232e-10, 9.285e-11, 6.941e-11, 4.951e-11,
                3.539e-11, 2.402e-11, 1.522e-11, 9.676e-12, 6.056e-12, 3.745e-12, 2.34e-12, 1.463e-12, 9.186e-13, 5.769e-13, 3.322e-13, 1.853e-13, 1.035e-13, 7.173e-14, 5.382e-14, 4.036e-14, 3.401e-14, 2.997e-14,
00532
00533
00534
                2.635e-14, 2.316e-14, 2.034e-14, 1.783e-14, 1.56e-14, 1.363e-14,
00536
                1.19e-14, 1.037e-14, 9.032e-15, 7.846e-15, 6.813e-15, 5.912e-15,
00537
                5.121e-15, 4.431e-15, 3.829e-15, 3.306e-15, 2.851e-15, 2.456e-15,
00538
                2.114e-15, 1.816e-15, 1.559e-15,
                                                                       1.337e-15, 1.146e-15, 9.811e-16,
                8.389e-16, 7.162e-16, 6.109e-16, 5.203e-16, 4.425e-16, 3.76e-16,
00539
                3.184e-16, 2.692e-16, 2.274e-16, 1.917e-16, 1.61e-16, 1.35e-16, 1.131e-16, 9.437e-17, 7.874e-17, 6.57e-17, 5.481e-17, 4.579e-17,
00540
00542
                3.828e-17, 3.204e-17, 2.691e-17, 2.264e-17, 1.912e-17, 1.626e-17,
00543
                1.382e-17, 1.174e-17, 9.972e-18, 8.603e-18, 7.45e-18, 6.453e-18,
00544
                5.623e-18, 4.944e-18, 4.361e-18, 3.859e-18, 3.443e-18, 3.096e-18,
00545
                2.788e-18, 2.528e-18, 2.293e-18, 2.099e-18, 1.929e-18, 1.773e-18,
00546
                1.64e-18
00547
00548
00549
             static double hocl[121] = {
00550
              1.056e-12, 1.194e-12, 1.35e-12, 1.531e-12, 1.737e-12, 1.982e-12,
               2.263e-12, 2.599e-12, 2.991e-12, 3.459e-12, 4.012e-12, 4.662e-12, 5.438e-12, 6.35e-12, 7.425e-12, 8.686e-12, 1.016e-11, 1.188e-11, 1.389e-11, 1.659e-11, 2.087e-11, 2.621e-11, 3.265e-11, 4.064e-11, 4.859e-11, 5.441e-11, 6.09e-11, 6.373e-11, 6.611e-11, 6.94e-11,
00551
00552
00553
00555
                7.44e-11, 7.97e-11, 8.775e-11, 9.722e-11, 1.064e-10, 1.089e-10
00556
                1.114e-10, 1.106e-10, 1.053e-10, 1.004e-10, 9.006e-11, 7.778e-11,
00557
                6.739 e-11, \; 5.636 e-11, \; 4.655 e-11, \; 3.845 e-11, \; 3.042 e-11, \; 2.368 e-11, \; 4.655 e-11, \; 
00558
                1.845e-11, 1.442e-11, 1.127e-11, 8.814e-12, 6.544e-12, 4.763e-12,
                3.449e-12, 2.612e-12, 1.999e-12, 1.526e-12, 1.16e-12, 8.793e-13,
00559
                6.655e-13, 5.017e-13, 3.778e-13, 2.829e-13, 2.117e-13, 1.582e-13,
                1.178e-13, 8.755e-14, 6.486e-14, 4.799e-14, 3.54e-14, 2.606e-14,
00561
00562
                1.916e-14, 1.403e-14, 1.026e-14, 7.48e-15, 5.446e-15, 3.961e-15,
00563
                2.872e-15, 2.076e-15, 1.498e-15, 1.077e-15, 7.726e-16, 5.528e-16,
                3.929e-16, 2.785e-16, 1.969e-16, 1.386e-16, 9.69e-17, 6.747e-17,
00564
00565
                4.692e-17, 3.236e-17, 2.232e-17, 1.539e-17, 1.061e-17, 7.332e-18,
                5.076e-18, 3.522e-18, 2.461e-18, 1.726e-18, 1.22e-18, 8.75e-19,
                6.264e-19, 4.482e-19, 3.207e-19, 2.368e-19, 1.762e-19, 1.312e-19, 9.891e-20, 7.595e-20, 5.87e-20, 4.567e-20, 3.612e-20, 2.904e-20,
00567
00568
00569
                2.343e-20, 1.917e-20, 1.568e-20, 1.308e-20, 1.1e-20, 9.25e-21,
00570
                7.881e-21
00571
00572
            static double n2o[121] =
              3.17e-07, 3.03e-07,
00574
00575
                2.984e-07, 2.938e-07, 2.892e-07, 2.847e-07, 2.779e-07, 2.705e-07, 2.631e-07, 2.557e-07, 2.484e-07, 2.345e-07, 2.201e-07, 2.01e-07,
00576
00577
00578
                1.754e-07, 1.532e-07, 1.329e-07, 1.154e-07, 1.003e-07, 8.735e-08,
                7.617e-08, 6.512e-08, 5.547e-08, 4.709e-08, 3.915e-08, 3.259e-08,
00580
                2.738e-08, 2.327e-08, 1.98e-08, 1.711e-08, 1.493e-08, 1.306e-08,
00581
                1.165e-08, 1.049e-08, 9.439e-09, 8.375e-09, 7.391e-09, 6.525e-09,
00582
                5.759e-09, 5.083e-09, 4.485e-09, 3.953e-09, 3.601e-09, 3.27e-09,
                2.975e-09, 2.757e-09, 2.556e-09, 2.37e-09, 2.195e-09, 2.032e-09, 1.912e-09, 1.79e-09, 1.679e-09, 1.572e-09, 1.482e-09, 1.402e-09,
00583
00584
                 1.326e-09, 1.254e-09, 1.187e-09, 1.127e-09, 1.071e-09, 1.02e-09,
                9.673e-10, 9.193e-10, 8.752e-10, 8.379e-10, 8.017e-10, 7.66e-10,
00586
00587
                7.319e-10, 7.004e-10, 6.721e-10, 6.459e-10, 6.199e-10, 5.942e-10,
00588
                5.703e-10, 5.488e-10, 5.283e-10, 5.082e-10, 4.877e-10, 4.696e-10,
                4.52e-10, 4.355e-10, 4.198e-10, 4.039e-10, 3.888e-10, 3.754e-10, 3.624e-10, 3.499e-10, 3.381e-10, 3.267e-10, 3.163e-10, 3.058e-10,
00589
00590
```

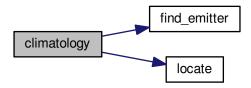
```
2.959e-10, 2.864e-10, 2.77e-10, 2.686e-10, 2.604e-10, 2.534e-10,
                           2.462e-10, 2.386e-10, 2.318e-10, 2.247e-10, 2.189e-10, 2.133e-10, 2.071e-10, 2.014e-10, 1.955e-10, 1.908e-10, 1.86e-10, 1.817e-10
00592
00593
00594
00595
00596
                      static double n2o5[121] = {
                         1.231e-11, 3.035e-12, 1.702e-12, 9.877e-13, 8.081e-13, 9.039e-13,
                            1.169e-12, 1.474e-12, 1.651e-12, 1.795e-12, 1.998e-12, 2.543e-12,
00598
00599
                            4.398e-12, 7.698e-12, 1.28e-11, 2.131e-11, 3.548e-11, 5.894e-11,
                           7.645e-11, 1.089e-10, 1.391e-10, 1.886e-10, 2.386e-10, 2.986e-10, 3.487e-10, 3.994e-10, 4.5e-10, 4.6e-10, 4.591e-10, 4.1e-10, 3.488e-10,
00600
00601
                            2.846e-10, 2.287e-10, 1.696e-10, 1.011e-10, 6.428e-11, 4.324e-11,
00602
                            2.225e-11, 6.214e-12, 3.608e-12, 8.793e-13, 4.491e-13, 1.04e-13,
00603
                            6.1e-14, 3.436e-14, 6.671e-15, 1.171e-15, 5.848e-16, 1.212e-16,
00604
00605
                            le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00606
                            1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00607
                            1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00608
                            1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
                            le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
00610
                            le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16, le-16,
                            1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16, 1e-16,
00611
00612
                           1e-16, 1e-16
00613
00614
                      static double nh3[121] = {
00615
                         le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10,
                            1e-10, 1e-10, 1e-10, 1e-10, 9.444e-11, 8.488e-11, 7.241e-11, 5.785e-11,
00617
00618
                            4.178e-11, 3.018e-11, 2.18e-11, 1.574e-11, 1.137e-11, 8.211e-12,
00619
                           5.973e-12, 4.327e-12, 3.118e-12, 2.234e-12, 1.573e-12, 1.04e-12,
00620
                            6.762e-13, 4.202e-13, 2.406e-13, 1.335e-13, 6.938e-14, 3.105e-14,
                           1.609e-14, 1.033e-14, 6.432e-15, 4.031e-15, 2.555e-15, 1.656e-15, 1.15e-15, 7.904e-16, 5.63e-16, 4.048e-16, 2.876e-16, 2.004e-16,
00621
00622
                            1.356e-16, 9.237e-17, 6.235e-17, 4.223e-17, 3.009e-17, 2.328e-17,
00623
00624
                            2.002e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                           1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 
00625
00626
                           1.914e-17, 
00627
00629
                            1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                            1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
00630
00631
                           1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17,
                           1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 1.914e-17, 
00632
00633
00634
                            1.914e-17
00636
00637
00638
                      static double no[121] = {
                           2.586e-10, 4.143e-11, 1.566e-11, 9.591e-12, 8.088e-12, 8.462e-12,
00639
                            1.013e-11, 1.328e-11, 1.855e-11, 2.678e-11, 3.926e-11, 5.464e-11,
00640
                            7.012e-11, 8.912e-11, 1.127e-10, 1.347e-10, 1.498e-10, 1.544e-10,
                           1.602e-10, 1.824e-10, 2.078e-10, 2.366e-10, 2.691e-10, 5.141e-10,
00642
00643
                           8.259e-10, 1.254e-09, 1.849e-09, 2.473e-09, 3.294e-09, 4.16e-09,
                           5.095e-09, 6.11e-09, 6.93e-09, 7.888e-09, 8.903e-09, 9.713e-09, 1.052e-08, 1.115e-08, 1.173e-08, 1.21e-08, 1.228e-08, 1.239e-08,
00644
00645
                            1.231e-08, 1.213e-08, 1.192e-08, 1.138e-08, 1.085e-08, 1.008e-08,
00646
                            9.224e-09, 8.389e-09, 7.262e-09, 6.278e-09, 5.335e-09, 4.388e-09,
                            3.589e-09, 2.761e-09, 2.129e-09, 1.633e-09, 1.243e-09, 9.681e-10,
00648
                           8.355e-10, 7.665e-10, 7.442e-10, 8.584e-10, 9.732e-10, 1.063e-09,
00649
00650
                           1.163e-09, 1.286e-09, 1.472e-09, 1.707e-09, 2.032e-09, 2.474e-09,
00651
                            2.977e-09, 3.506e-09, 4.102e-09, 5.013e-09, 6.493e-09, 8.414e-09,
00652
                            1.077e-08, 1.367e-08, 1.777e-08, 2.625e-08, 3.926e-08, 5.545e-08,
                             7.195e-08, 9.464e-08, 1.404e-07, 2.183e-07, 3.329e-07, 4.535e-07,
                             6.158e-07, 8.187e-07, 1.075e-06, 1.422e-06, 1.979e-06, 2.71e-06,
00655
                           3.58e-06, 4.573e-06, 5.951e-06, 7.999e-06, 1.072e-05, 1.372e-05,
00656
                           1.697e-05, 2.112e-05, 2.643e-05, 3.288e-05, 3.994e-05, 4.794e-05,
                            5.606e-05, 6.383e-05, 7.286e-05, 8.156e-05, 8.883e-05, 9.469e-05, 9.848e-05, 0.0001023, 0.0001066, 0.0001115, 0.0001145, 0.0001142,
00657
00658
00659
                           0.0001133
00661
00662
                      static double no2[121] = {
                           3.036e-09, 2.945e-10, 9.982e-11, 5.069e-11, 3.485e-11, 2.982e-11,
00663
                            2.947e-11, 3.164e-11, 3.714e-11, 4.586e-11, 6.164e-11, 8.041e-11, 9.982e-11, 1.283e-10, 1.73e-10, 2.56e-10, 3.909e-10, 5.959e-10,
00664
00665
                            9.081e-10, 1.384e-09, 1.788e-09, 2.189e-09, 2.686e-09, 3.091e-09,
                            3.49e-09, 3.796e-09, 4.2e-09, 5.103e-09, 6.005e-09, 6.3e-09, 6.706e-09,
00667
00668
                            7.07e-09, 7.434e-09, 7.663e-09, 7.788e-09, 7.8e-09, 7.597e-09,
                           7.482e-09, 7.227e-09, 6.403e-09, 5.585e-09, 4.606e-09, 3.703e-09, 2.984e-09, 2.183e-09, 1.48e-09, 8.441e-10, 5.994e-10, 3.799e-10,
00669
00670
00671
                            2.751e-10, 1.927e-10, 1.507e-10, 1.102e-10, 6.971e-11, 5.839e-11,
                            3.904e-11, 3.087e-11, 2.176e-11, 1.464e-11, 1.209e-11, 8.497e-12,
                            6.477e-12, 4.371e-12, 2.914e-12, 2.424e-12, 1.753e-12, 1.35e-12,
00673
00674
                            9.417e-13, 6.622e-13, 5.148e-13, 3.841e-13, 3.446e-13, 3.01e-13,
00675
                           2.551e-13, 2.151e-13, 1.829e-13, 1.64e-13, 1.475e-13, 1.352e-13,
                            1.155e-13, 9.963e-14, 9.771e-14, 9.577e-14, 9.384e-14, 9.186e-14, 9e-14, 9e-14,
00676
00677
```

```
9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14,
                    9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14, 9e-14
00679
00680
00681
00682
00683
                static double o3[121] = {
                    2.218e-08, 3.394e-08, 3.869e-08, 4.219e-08, 4.501e-08, 4.778e-08,
                     5.067e-08, 5.402e-08, 5.872e-08, 6.521e-08, 7.709e-08, 9.461e-08,
00685
00686
                     1.269e-07, 1.853e-07, 2.723e-07, 3.964e-07, 5.773e-07, 8.2e-07,
                    1.155e-06, 1.59e-06, 2.076e-06, 2.706e-06, 3.249e-06, 3.848e-06, 4.459e-06, 4.986e-06, 5.573e-06, 5.958e-06, 6.328e-06, 6.661e-06, 6.9e-06, 7.146e-06, 7.276e-06, 7.374e-06, 7.447e-06, 7.383e-06,
00687
00688
00689
00690
                      7.321e-06, 7.161e-06, 6.879e-06, 6.611e-06, 6.216e-06, 5.765e-06,
                     5.355e-06, 4.905e-06, 4.471e-06, 4.075e-06, 3.728e-06, 3.413e-06,
00691
00692
                     3.125e-06, 2.856e-06, 2.607e-06, 2.379e-06, 2.17e-06, 1.978e-06,
                     1.8e-06, 1.646e-06, 1.506e-06, 1.376e-06, 1.233e-06, 1.102e-06, 9.839e-07, 8.771e-07, 7.814e-07, 6.947e-07, 6.102e-07, 5.228e-07, 4.509e-07, 3.922e-07, 3.501e-07, 3.183e-07, 2.909e-07, 2.686e-07,
00693
00694
00695
                     2.476e-07, 2.284e-07, 2.109e-07, 2.003e-07, 2.013e-07, 2.022e-07,
                     2.032e-07, 2.042e-07, 2.097e-07, 2.361e-07, 2.656e-07,
00697
                     3.37e-07, 3.826e-07, 4.489e-07, 5.26e-07, 6.189e-07, 7.312e-07, 8.496e-07, 8.444e-07, 8.392e-07, 8.339e-07, 8.286e-07, 8.234e-07,
00698
00699
                    8.181e-07, 8.129e-07, 8.077e-07, 8.026e-07, 6.918e-07, 5.176e-07, 3.865e-07, 2.885e-07, 2.156e-07, 1.619e-07, 1.219e-07, 9.161e-08, 6.972e-08, 5.399e-08, 3.498e-08, 2.111e-08, 1.322e-08, 8.482e-09,
00700
00701
00702
                     5.527e-09, 3.423e-09, 2.071e-09, 1.314e-09, 8.529e-10, 5.503e-10,
00703
00704
                    3.665e-10
00705
                };
00706
00707
                static double ocs[121] = {
00708
                   6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 6e-10, 5.997e-10,
                     5.989e-10, 5.881e-10, 5.765e-10, 5.433e-10, 5.074e-10, 4.567e-10,
00710
                     4.067e-10, 3.601e-10, 3.093e-10, 2.619e-10, 2.232e-10,
                                                                                                                                          1.805e-10
00711
                     1.46 e^{-10}, \ 1.187 e^{-10}, \ 8.03 e^{-11}, \ 5.435 e^{-11}, \ 3.686 e^{-11}, \ 2.217 e^{-11},
00712
                     1.341e-11, 8.756e-12, 4.511e-12, 2.37e-12, 1.264e-12, 8.28e-13,
                     5.263e-13, 3.209e-13, 1.717e-13, 9.068e-14, 4.709e-14, 2.389e-14,
00713
00714
                     1.236e-14, 1.127e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00716
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00717
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00718
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00719
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 
00720
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00722
00723
                     1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
00724
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14,
                    1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14, 1.091e-14
00725
00726
00727
00728
00729
00730
                static double sf6[121] = {
                  4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12, 4.103e-12,
00731
                     4.103e-12, 4.103e-12, 4.103e-12, 4.087e-12, 4.064e-12, 4.023e-12, 3.988e-12, 3.941e-12, 3.884e-12, 3.755e-12, 3.622e-12, 3.484e-12, 3.32e-12, 3.144e-12, 2.978e-12, 2.811e-12, 2.653e-12, 2.489e-12,
00732
00733
00735
                     2.332e-12, 2.199e-12, 2.089e-12, 2.013e-12, 1.953e-12, 1.898e-12,
00736
                    1.859e-12, 1.826e-12, 1.798e-12, 1.776e-12, 1.757e-12, 1.742e-12,
00737
                    1.728e-12, 1.717e-12, 1.707e-12, 1.698e-12, 1.691e-12, 1.685e-12,
                     1.679e-12, 1.675e-12, 1.671e-12, 1.668e-12, 1.665e-12, 1.663e-12,
00738
00739
                    1.661e-12, 1.659e-12, 1.658e-12, 1.657e-12, 1.656e-12, 1.655e-12,
                     1.654e-12, 1.653e-12, 1.653e-12, 1.652e-12, 1.652e-12, 1.652e-12,
00741
                     1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12, 1.651e-12,
00742
                     1.651e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00743
                    1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00744
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00745
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00747
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00748
                     1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12,
00749
                    1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12, 1.65e-12
00750
                };
00751
00752
                static double so2[121] = {
                   le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10, le-10,
00753
00754
                     1e-10, 1e-10, 9.867e-11, 9.537e-11, 9e-11, 8.404e-11, 7.799e-11,
00755
                     7.205e-11, 6.616e-11, 6.036e-11, 5.475e-11, 5.007e-11, 4.638e-11,
00756
                     4.346e-11, 4.055e-11, 3.763e-11, 3.471e-11, 3.186e-11, 2.905e-11,
                    2.631e-11, 2.358e-11, 2.415e-11, 2.949e-11, 3.952e-11, 5.155e-11, 6.76e-11, 8.741e-11, 1.099e-10, 1.278e-10, 1.414e-10, 1.512e-10, 1.607e-10, 1.699e-10, 1.774e-10, 1.832e-10, 1.871e-10, 1.907e-10,
00757
00758
00760
                     1.943e-10, 1.974e-10, 1.993e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00761
                     2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
00762
                     2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
                     2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e
00763
00764
```

```
2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10,
          2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10, 2e-10
00766
00767
00768
00769
00770
        static int ig co2 = -999;
00771
00772
        double co2, *q[NG] = {NULL};
00773
00774
        int ig, ip, iw, iz;
00775
00776
        /* Find emitter index of CO2... */
00777
        if (ig_co2 == -999)
00778
          ig_co2 = find_emitter(ct1, "CO2");
00779
00780
        /* Identify variable... */
        for (ig = 0; ig < ctl->ng; ig++) {
   q[ig] = NULL;
00781
00782
          if (strcasecmp(ctl->emitter[ig], "C2H2") == 0)
00784
            q[ig] = c2h2;
00785
          if (strcasecmp(ctl->emitter[ig], "C2H6") == 0)
            q[ig] = c2h6;
00786
00787
          if (strcasecmp(ctl->emitter[ig], "CC14") == 0)
00788
            q[ig] = ccl4;
00789
          if (strcasecmp(ctl->emitter[iq], "CH4") == 0)
00790
            q[ig] = ch4;
              (strcasecmp(ctl->emitter[ig], "ClO") == 0)
00791
          if
            q[ig] = clo;
00792
          if (strcasecmp(ctl->emitter[ig], "ClONO2") == 0)
00793
00794
            q[ig] = clono2;
          if (strcasecmp(ctl->emitter[ig], "CO") == 0)
00795
00796
            q[ig] = co;
00797
             (strcasecmp(ctl->emitter[ig], "COF2") == 0)
            q[ig] = cof2;
00798
00799
          if (strcasecmp(ctl->emitter[ig], "F11") == 0)
00800
            q[ig] = f11;
00801
          if (strcasecmp(ctl->emitter[iq], "F12") == 0)
            q[ig] = f12;
00803
          if (strcasecmp(ctl->emitter[ig], "F14") == 0)
00804
            q[ig] = f14;
          if (strcasecmp(ctl->emitter[ig], "F22") == 0)
00805
00806
            q[ig] = f22;
          if (strcasecmp(ctl->emitter[ig], "H2O") == 0)
00807
80800
            q[ig] = h2o;
          if (strcasecmp(ctl->emitter[ig], "H2O2") == 0)
00809
            q[ig] = h2o2;
00810
00811
          if (strcasecmp(ctl->emitter[ig], "HCN") == 0)
00812
            q[ig] = hcn;
          if (strcasecmp(ctl->emitter[ig], "HNO3") == 0)
00813
            q[ig] = hno3;
00814
00815
             (strcasecmp(ctl->emitter[ig], "HNO4") == 0)
00816
            q[ig] = hno4;
00817
          if (strcasecmp(ctl->emitter[ig], "HOC1") == 0)
          q[ig] = hocl;
if (strcasecmp(ctl->emitter[ig], "N2O") == 0)
00818
00819
            q[ig] = n20;
00820
           if (strcasecmp(ctl->emitter[ig], "N205") == 0)
            q[ig] = n2o5;
00822
00823
          if (strcasecmp(ctl->emitter[ig], "NH3") == 0)
00824
            q[ig] = nh3;
00825
          if (strcasecmp(ctl->emitter[iq], "NO") == 0)
00826
            q[ig] = no;
00827
          if (strcasecmp(ctl->emitter[ig], "NO2") == 0)
            q[ig] = no2;
00828
00829
          if (strcasecmp(ctl->emitter[ig], "03") == 0)
            q[ig] = 03;
00830
          if (strcasecmp(ctl->emitter[ig], "OCS") == 0)
00831
00832
            q[iq] = ocs;
          if (strcasecmp(ctl->emitter[ig], "SF6") == 0)
00833
            q[ig] = sf6;
00835
              (strcasecmp(ctl->emitter[ig], "SO2") == 0)
00836
            q[ig] = so2;
00837
00838
00839
        /* Loop over atmospheric data points... */
        for (ip = 0; ip < atm->np; ip++) {
00840
00841
00842
           /\star Get altitude index... \star/
00843
          iz = locate(z, 121, atm->z[ip]);
00844
00845
          /* Interpolate pressure... */
00846
          atm \rightarrow p[ip] = EXP(z[iz], pre[iz], z[iz + 1], pre[iz + 1], atm \rightarrow z[ip]);
00847
00848
          /* Interpolate temperature... */
00849
          atm \rightarrow t[ip] = LIN(z[iz], tem[iz], z[iz + 1], tem[iz + 1], atm \rightarrow z[ip]);
00850
00851
          /* Interpolate trace gases... */
```

```
for (ig = 0; ig < ctl->ng; ig++)
           if (q[ig] != NULL)
00854
              atm->q[ig][ip] =
00855
               LIN(z[iz], q[ig][iz], z[iz + 1], q[ig][iz + 1], atm->z[ip]);
00856
00857
              atm->q[iq][ip] = 0;
00859
          /* Set CO2... *
00860
          if (ig_co2 >= 0) {
00861
            co2 =
              371.789948e-6 + 2.026214e-6 * (atm->time[ip] - 63158400.) / 31557600.;
00862
            atm->q[ig\_co2][ip] = co2;
00863
00864
00865
00866
          /* Set extinction to zero... */
00867
          for (iw = 0; iw < ctl->nw; iw++)
            atm->k[iw][ip] = 0;
00868
00869
00870 }
```

Here is the call graph for this function:



## 5.15.2.6 double ctmco2 ( double nu, double p, double t, double u)

Compute carbon dioxide continuum (optical depth).

Definition at line 874 of file jurassic.c.

```
00878
00879
              static double co2296[2001] = { 9.3388e-5, 9.7711e-5, 1.0224e-4, 1.0697e-4,
00880
                  1.1193e-4, 1.1712e-4, 1.2255e-4, 1.2824e-4, 1.3419e-4, 1.4043e-4,
                   1.4695e-4, 1.5378e-4, 1.6094e-4, 1.6842e-4, 1.7626e-4, 1.8447e-4,
                  1.9307e-4, 2.0207e-4, 2.1149e-4, 2.2136e-4, 2.3169e-4, 2.4251e-4, 2.5384e-4, 2.657e-4, 2.7813e-4, 2.9114e-4, 3.0477e-4, 3.1904e-4,
00883
00884
                  3.3399e-4, 3.4965e-4, 3.6604e-4, 3.8322e-4, 4.0121e-4, 4.2006e-4, 4.398e-4, 4.6047e-4, 4.8214e-4, 5.0483e-4, 5.286e-4, 5.535e-4, 5.7959e-4, 6.0693e-4, 6.3557e-4, 6.6558e-4, 6.9702e-4, 7.2996e-4,
00885
00886
00887
                  7.6449e-4, 8.0066e-4, 8.3856e-4, 8.7829e-4, 9.1991e-4,
                  .0010093, .0010572, .0011074, .00116, .0012152, .001273, .0013336, .0013972, .0014638, .0015336, .0016068, .0016835,
00889
00890
                  .001764, .0018483, .0019367, .0020295, .0021267, .0022286, .0023355, .0024476, .0025652, .0026885, .0028178, .0029534,
00891
00892
00893
                   .0030956, .0032448, .0034012, .0035654, .0037375, .0039181,
                  .0041076, .0043063, .0045148, .0047336, .0049632, .005204, .0054567, .0057219, .0060002, .0062923, .0065988, .0069204,
00894
00895
                  .007258, .0076123, .0079842, .0083746, .0087844, .0092146, .0096663, .01014, .010638, .011161, .01171, .012286, .012891,
00896
00897
                   .013527, .014194, .014895, .015631, .016404, .017217, .01807,
00898
                  .018966, .019908, .020897, .021936, .023028, .024176, .025382, .026649, .027981, .02938, .030851, .032397, .034023, .035732, .037528, .039416, .041402, .04349, .045685, .047994, .050422, .052975, .055661, .058486, .061458, .064584, .067873, .071334,
00899
00900
00901
00902
                  .074975, .078807, .082839, .087082, .091549, .09649, .1012, .10641, .11189, .11767, .12375, .13015, .13689, .14399, .15147, .15935, .16765, .17639, .18561, .19531, .20554, .21632, .22769, .23967, .25229, .2656, .27964, .29443, .31004, .3265, .34386, .36218, .3815, .40188, .42339, .44609, .47004, .49533, .52202,
00903
00904
00905
00906
```

```
.5502, .57995, .61137, .64455, .6796, .71663, .75574, .79707,
               .84075, .88691, .9357, .98728, 1.0418, 1.0995, 1.1605, 1.225, 1.2932, 1.3654, 1.4418, 1.5227, 1.6083, 1.6989, 1.7948, 1.8964, 2.004, 2.118, 2.2388, 2.3668, 2.5025, 2.6463, 2.7988, 2.9606,
00909
00910
00911
00912
               3.1321, 3.314, 3.5071, 3.712, 3.9296, 4.1605, 4.4058, 4.6663, 4.9431, 5.2374, 5.5501, 5.8818, 6.2353, 6.6114, 7.0115, 7.4372,
00913
               7.8905, 8.3731, 8.8871, 9.4349, 10.019, 10.641, 11.305, 12.013,
               12.769, 13.576, 14.437, 15.358, 16.342, 17.39, 18.513, 19.716,
00915
00916
               21.003, 22.379, 23.854, 25.436, 27.126, 28.942, 30.89, 32.973,
              35.219, 37.634, 40.224, 43.021, 46.037, 49.29, 52.803, 56.447, 60.418, 64.792, 69.526, 74.637, 80.182, 86.193, 92.713, 99.786, 107.47, 115.84, 124.94, 134.86, 145.69, 157.49, 170.3, 184.39, 199.83, 216.4, 234.55, 254.72, 276.82, 299.85, 326.16, 354.99, 386.51, 416.68, 449.89, 490.12, 534.35, 578.25, 632.26, 692.61,
00917
00918
00919
00920
00921
00922
               756.43, 834.75, 924.11, 1016.9, 996.96, 1102.7, 1219.2, 1351.9,
               1494.3, 1654.1, 1826.5, 2027.9, 2249., 2453.8, 2714.4, 2999.4, 3209.5, 3509., 3840.4, 3907.5, 4190.7, 4533.5, 4648.3, 5059.1, 5561.6, 6191.4, 6820.8, 7905.9, 9362.2, 2431.3, 2211.3, 2046.8,
00923
00924
00925
               2023.8, 1985.9, 1905.9, 1491.1, 1369.8, 1262.2, 1200.7, 887.74,
               820.25, 885.23, 887.21, 816.73, 1126.9, 1216.2, 1272.4, 1579.5,
               1634.2, 1656.3, 1657.9, 1789.5, 1670.8, 1509.5, 8474.6, 7489.2, 6793.6, 6117., 5574.1, 5141.2, 5084.6, 4745.1, 4413.2, 4102.8,
00928
00929
               4024.7, 3715., 3398.6, 3100.8, 2900.4, 2629.2, 2374., 2144.7, 1955.8, 1760.8, 1591.2, 1435.2, 1296.2, 1174., 1065.1, 967.76, 999.48, 897.45, 809.23, 732.77, 670.26, 611.93, 560.11, 518.77,
00930
00931
00932
               476.84, 438.8, 408.48, 380.21, 349.24, 322.71, 296.65, 272.85,
               251.96, 232.04, 213.88, 197.69, 182.41, 168.41, 155.79, 144.05,
00934
              133.31, 123.48, 114.5, 106.21, 98.591, 91.612, 85.156, 79.204, 73.719, 68.666, 63.975, 59.637, 56.35, 52.545, 49.042, 45.788, 42.78, 39.992, 37.441, 35.037, 32.8, 30.744, 28.801, 26.986, 25.297, 23.731, 22.258, 20.883, 19.603, 18.403, 17.295, 16.249,
00935
00936
00937
00938
00939
               15.271, 14.356, 13.501, 12.701, 11.954, 11.254, 10.6, 9.9864,
               9.4118, 8.8745, 8.3714, 7.8997, 7.4578, 7.0446, 6.6573, 6.2949,
00940
00941
               5.9577, 5.6395, 5.3419, 5.063, 4.8037, 4.5608, 4.3452, 4.1364,
               3.9413, 3.7394, 3.562, 3.3932, 3.2325, 3.0789, 2.9318, 2.7898, 2.6537, 2.5225, 2.3958, 2.2305, 2.1215, 2.0245, 1.9427, 1.8795
00942
00943
00944
               1.8336, 1.7604, 1.7016, 1.6419, 1.5282, 1.4611, 1.3443, 1.27,
               1.1675, 1.0824, 1.0534, .99833, .95854, .92981, .90887, .89346,
00946
               .88113, .87068, .86102, .85096, .88262, .86151, .83565, .80518,
               .77045, .73736, .74744, .74954, .75773, .82267, .83493, .89402,
00947
00948
               .89725, .93426, .95564, .94045, .94174, .93404, .92035, .90456
               .88621, .86673, .78117, .7515, .72056, .68822, .65658, .62764,
00949
               .55984, .55598, .57407, .60963, .63763, .66198, .61132, .60972, .52496, .50649, .41872, .3964, .32422, .27276, .24048, .23772,
00950
00951
               .2286, .22711, .23999, .32038, .34371, .36621, .38561, .39953,
00953
               .40636, .44913, .42716, .3919, .35477, .33935, .3351, .39746,
00954
               .40993, .49398, .49956, .56157, .54742, .57295, .57386, .55417,
              . 50745, .471, .43446, .39102, .34993, .31269, .27888, .24912, .22291, .19994, .17972, .16197, .14633, .13252, .12029, .10942, .099745, .091118, .083404, .076494, .070292, .064716, .059697, .055173, .051093, .047411, .044089, .041092, .038392, .035965,
00955
00956
00957
               .033789, .031846, .030122, .028607, .02729, .026169, .025209,
00959
00960
               .024405, .023766, .023288, .022925, .022716, .022681, .022685,
               .022768, .023133, .023325, .023486, .024004, .024126, .024083, .023785, .024023, .023029, .021649, .021108, .019454, .017809,
00961
00962
              .017292, .016635, .017037, .018068, .018977, .018756, .017847, .016557, .016142, .014459, .012869, .012381, .010875, .0098701, .009285, .0091698, .0091701, .0096145, .010553, .01106, .012613,
00963
00965
               .014362, .015017, .016507, .017741, .01768, .017784, .0171, .016357, .016172, .017257, .018978, .020935, .021741, .023567,
00966
00967
00968
               .025183, .025589, .026732, .027648, .028278, .028215, .02856,
              .029163, .029062, .028851, .028497, .027825, .027801, .026523, .02487, .022967, .022168, .020194, .018605, .017903, .018439, .019697, .020311, .020855, .020057, .018608, .016738, .015963, .013844, .011801, .011134, .0097573, .0086007, .0086226,
00969
00971
00972
00973
               .0083721, .0090978, .0097616, .0098426, .011317, .012853,
                                                                                                         .01447.
               .014657, .015771, .016351, .016079, .014829, .013431, .013185, .013207, .01448, .016176, .017971, .018265, .019526, .020455, .019797, .019802, .0194, .018176, .017505, .016197, .015339,
00974
00975
               .014401, .013213, .012203, .011186, .010236, .0093288,
               .0076837, .0069375, .0062614, .0056628, .0051153, .0046015,
00978
00979
               .0041501, .003752, .0033996, .0030865, .0028077, .0025586,
               .0023355, .0021353, .0019553, .0017931, .0016466, .0015141, .0013941, .0012852, .0011862, .0010962, .0010142, 9.3935e-4, 8.71e-4, 8.0851e-4, 7.5132e-4, 6.9894e-4, 6.5093e-4, 6.0689e-4,
00980
00981
00982
               5.6647e-4, 5.2935e-4, 4.9525e-4, 4.6391e-4, 4.3509e-4, 4.086e-4,
               3.8424e-4, 3.6185e-4, 3.4126e-4, 3.2235e-4, 3.0498e-4, 2.8904e-4,
00984
00985
               2.7444e-4, 2.6106e-4, 2.4883e-4, 2.3766e-4, 2.275e-4, 2.1827e-4,
00986
               2.0992e-4, 2.0239e-4, 1.9563e-4, 1.896e-4, 1.8427e-4, 1.796e-4,
               1.7555e-4, 1.7209e-4, 1.692e-4, 1.6687e-4, 1.6505e-4, 1.6375e-4,
00987
               1.6294e-4, 1.6261e-4, 1.6274e-4, 1.6334e-4, 1.6438e-4, 1.6587e-4,
00988
               1.678e-4, 1.7017e-4, 1.7297e-4, 1.762e-4, 1.7988e-4, 1.8399e-4,
               1.8855e-4, 1.9355e-4, 1.9902e-4, 2.0494e-4, 2.1134e-4, 2.1823e-4,
00990
00991
               2.2561e-4, 2.335e-4, 2.4192e-4, 2.5088e-4, 2.604e-4, 2.705e-4,
00992
               2.8119e-4, 2.9251e-4, 3.0447e-4, 3.171e-4, 3.3042e-4, 3.4447e-4,
               3.5927e-4, 3.7486e-4, 3.9127e-4, 4.0854e-4, 4.267e-4, 4.4579e-4, 4.6586e-4, 4.8696e-4, 5.0912e-4, 5.324e-4, 5.5685e-4, 5.8253e-4,
00993
00994
```

```
6.0949e-4, 6.378e-4, 6.6753e-4, 6.9873e-4, 7.3149e-4, 7.6588e-4,
                   8.0198e-4, 8.3987e-4, 8.7964e-4, 9.2139e-4, 9.6522e-4, .0010112,
00996
00997
                   .0010595, .0011102, .0011634, .0012193, .001278, .0013396,
00998
                    .0014043, .0014722, .0015436, .0016185, .0016972, .0017799,
                   .0018668, .001958, .0020539, .0021547, .0022606, .0023719, .002489, .002612, .0027414, .0028775, .0030206, .0031712, .0033295, .0034962, .0036716, .0038563, .0040506, .0042553,
00999
01000
                    .0044709, .004698, .0049373, .0051894, .0054552, .0057354,
01002
                   .006031, .0063427, .0066717, .0070188, .0073854, .0077726, .0081816, .0086138, .0090709, .0095543, .010066, .010607, .011181, .011789, .012433, .013116, .013842, .014613, .015432,
01003
01004
01005
                   .016304, .017233, .018224, .019281, .020394, .021574, .022836, .024181, .025594, .027088, .028707, .030401, .032245, .034219, .036262, .038539, .040987, .043578, .04641, .04949, .052726, .056326, .0602, .064093, .068521, .073278, .077734, .083064,
01006
01008
01009
                   .088731, .093885, .1003, .1072, .11365, .12187, .13078, .13989, .15095, .16299, .17634, .19116, .20628, .22419, .24386, .26587,
01010
01011
                   . 28811, .31399, .34321, .36606, .39675, .42742, .44243, .47197, .49993, .49027, .51147, .52803, .48931, .49729, .5026, .43854,
01012
                   .441, .44766, .43414, .46151, .50029, .55247, .43855, .32115, .32607, .3431, .36119, .38029, .41179, .43996, .47144, .51853, .55362, .59122, .66338, .69877, .74001, .82923, .86907, .90361,
01014
01015
01016
                   1.0025, 1.031, 1.0559, 1.104, 1.1178, 1.1341, 1.1547, 1.351, 1.4772, 1.4812, 1.4907, 1.512, 1.5442, 1.5853, 1.6358, 1.6963, 1.7674, 1.8474, 1.9353, 2.0335, 2.143, 2.2592, 2.3853, 2.5217,
01017
01018
                   2.6686, 2.8273, 2.9998, 3.183, 3.3868, 3.6109, 3.8564, 4.1159,
                   4.4079, 4.7278, 5.0497, 5.3695, 5.758, 6.0834, 6.4976, 6.9312,
01021
01022
                   7.38, 7.5746, 7.9833, 8.3791, 8.3956, 8.7501, 9.1067, 9.072,
                   9.4649, 9.9112, 10.402, 10.829, 11.605, 12.54, 12.713, 10.443, 10.825, 11.375, 11.955, 12.623, 13.326, 14.101, 15.041, 15.547, 16.461, 17.439, 18.716, 19.84, 21.036, 22.642, 23.901, 25.244, 27.03, 28.411, 29.871, 31.403, 33.147, 34.744, 36.456, 39.239,
01023
01024
01025
                   43.605, 45.162, 47.004, 49.093, 51.391, 53.946, 56.673, 59.629, 63.167, 66.576, 70.254, 74.222, 78.477, 83.034, 87.914, 93.18,
01027
01028
01029
                   98.77, 104.74, 111.15, 117.95, 125.23, 133.01, 141.33, 150.21,
                   159.71, 169.89, 180.93, 192.54, 204.99, 218.34, 232.65, 248., 264.47, 282.14, 301.13, 321.53, 343.48, 367.08, 392.5, 419.88, 449.4, 481.26, 515.64, 552.79, 592.99, 636.48, 683.61, 734.65,
01030
01031
01033
                   789.99, 850.02, 915.14, 985.81, 1062.5, 1147.1, 1237.8, 1336.4,
                    1443.2, 1558.9, 1684.2, 1819.2, 1965.2, 2122.6, 2291.7, 2470.8,
01034
                   2665.7, 2874.9, 3099.4, 3337.9, 3541., 3813.3, 4111.9, 4439.3, 4798.9, 5196., 5639.2, 6087.5, 6657.7, 7306.7, 8040.7, 8845.5,
01035
01036
                   9702.2, 10670., 11739., 12842., 14141., 15498., 17068., 18729., 20557., 22559., 25248., 27664., 30207., 32915., 35611., 38081.,
01037
                   40715., 43191., 41651., 42750., 43785., 44353., 44366., 44189., 43618., 42862., 41878., 35133., 35215., 36383., 39420., 44055.,
01040
01041
                   44155., 45850., 46853., 39197., 38274., 29942., 28553., 21792.,
01042
                   21228., 17106., 14955., 18181., 19557., 21427., 23728., 26301.,
                   28584., 30775., 32536., 33867., 40089., 39204., 37329., 34452., 31373., 33921., 34800., 36043., 44415., 45162., 52181., 50895., 54140., 50840., 50468., 48302., 44915., 40910., 36754., 32755.,
01043
01044
                   29093., 25860., 22962., 20448., 18247., 16326., 14645., 13165., 11861., 10708., 9686.9, 8779.7, 7971.9, 7250.8, 6605.7, 6027.2,
01046
01047
                   5507.3, 5039.1, 4616.6, 4234.8, 3889., 3575.4, 3290.5, 3031.3, 2795.2, 2579.9, 2383.1, 2203.3, 2038.6, 1887.6, 1749.1, 1621.9, 1505., 1397.4, 1298.3, 1207., 1122.8, 1045., 973.1, 906.64, 845.16, 788.22, 735.48, 686.57, 641.21, 599.1, 559.99, 523.64, 489.85, 458.42, 429.16, 401.92, 376.54, 352.88, 330.82, 310.24,
01048
01049
01050
01052
                   291.03, 273.09, 256.34, 240.69, 226.05, 212.37, 199.57, 187.59
01053
                   176.37, 165.87, 156.03, 146.82, 138.17, 130.07, 122.47, 115.34, 108.65, 102.37, 96.473, 90.934, 85.73, 80.84, 76.243, 71.922, 67.858, 64.034, 60.438, 57.052, 53.866, 50.866, 48.04, 45.379, 42.872, 40.51, 38.285, 36.188, 34.211, 32.347, 30.588, 28.929,
01054
01055
01056
                   27.362, 25.884, 24.489, 23.171, 21.929, 20.755, 19.646, 18.599, 17.61, 16.677, 15.795, 14.961, 14.174, 13.43, 12.725, 12.06,
01059
01060
                   11.431, 10.834, 10.27, 9.7361, 9.2302, 8.7518, 8.2997, 7.8724,
                   7.4674, 7.0848, 6.7226, 6.3794, 6.054, 5.745, 5.4525, 5.1752, 4.9121, 4.6625, 4.4259, 4.2015, 3.9888, 3.7872, 3.5961, 3.4149, 3.2431, 3.0802, 2.9257, 2.7792, 2.6402, 2.5084, 2.3834, 2.2648, 2.1522, 2.0455, 1.9441, 1.848, 1.7567, 1.6701, 1.5878, 1.5097,
01061
01062
01063
                   1.4356, 1.3651, 1.2981, 1.2345, 1.174, 1.1167, 1.062, 1.0101,
01065
01066
                   .96087, .91414, .86986, .82781, .78777, .74971, .71339, .67882,
01067
                    .64604, .61473, .58507, .55676, .52987, .5044, .48014, .45715,
                   .43527, .41453, .3948, .37609, .35831, .34142, .32524, .30995, .29536, .28142, .26807, .25527, .24311, .23166, .22077, .21053, .20081, .19143, .18261, .17407, .16603, .15833, .15089, .14385, .13707, .13065, .12449, .11865, .11306, .10774, .10266, .097818
01068
01069
01071
                   .093203, .088815, .084641, .080671, .076892, .073296, .069873, .066613, .06351, .060555, .05774, .055058, .052504, .050071, .047752, .045543, .043438, .041432, .039521, .037699, .035962,
01072
01073
01074
01075
                   .034307, .032729, .031225, .029791, .028423, .02712, .025877, .024692, .023563, .022485, .021458, .020478, .019543, .018652,
                   .017802, .016992, .016219, .015481, .014778, .014107, .013467, .012856, .012274, .011718, .011188, .010682, .0102, .0097393,
01077
01078
                   .0093001, .008881, .0084812, .0080997, .0077358, .0073885, .0070571, .0067409, .0064393, .0061514, .0058768, .0056147, .0053647, .0051262, .0048987, .0046816, .0044745, .0042769,
01079
01080
01081
```

```
.0040884, .0039088, .0037373, .0035739, .003418, .0032693,
               .0031277, .0029926, .0028639, .0027413, .0026245, .0025133, .0024074, .0023066, .0022108, .0021196, .002033, .0019507,
01084
01085
                .0018726, .0017985, .0017282, .0016617, .0015988, .0015394,
01086
                .0014834, .0014306, .0013811, .0013346, .0012911, .0012506,
                .0012131, .0011784, .0011465, .0011175, .0010912, .0010678, .0010472, .0010295, .0010147, .001003, 9.9428e-4, 9.8883e-4
01087
                9.8673e-4, 9.8821e-4, 9.9343e-4, .0010027, .0010164, .0010348,
01089
01090
                .0010586, .0010882, .0011245, .0011685, .0012145, .0012666,
                .0013095, .0013688, .0014048, .0014663, .0015309, .0015499, .0016144, .0016312, .001705, .0017892, .0018499, .0019715,
01091
01092
               .0021102, .0022442, .0024284, .002589, .0016763, .0029445, .0031193, .003346, .0034552, .0036906, .0037584, .0040084, .0041934, .0044587, .0047093, .0049759, .0053421, .0055134, .0059048, .0058663, .0061036, .0063259, .0059657, .0060653,
01093
01094
01095
01096
01097
                .0060972, .0055539, .0055653, .0055772, .005331, .0054953,
01098
                .0055919, .0058684, .006183, .0066675, .0069808, .0075142,
                .0078536, .0084282, .0089454, .0094625, .0093703, .0095857, .0099283, .010063, .010521, .0097778, .0098175, .010379, .010447,
01099
                .0105, .010617, .010706, .01078, .011177, .011212, .011304,
                .011446, .011603, .011816, .012165, .012545, .013069, .013539, .01411, .014776, .016103, .017016, .017994, .018978, .01998,
01102
01103
                .021799, .022745, .023681, .024627, .025562, .026992, .027958,
01104
               029013, 030154, 031402, 03228, 033651, 035272, 037088, 039021, 041213, 043597, 045977, 04877, 051809, 054943, 058064, 061528, 06537, 069309, 071928, 075752, 079589, 083352, 084096, 087497, 090817, 091198, 094966, 099045
01105
01106
01108
01109
                .10429, .10867, .11518, .12269, .13126, .14087, .15161, .16388,
01110
                .16423, .1759, .18721, .19994, .21275, .22513, .23041, .24231,
               25299, 25396, 26396, 27696, 27929, 2908, 30595, 31433, 3282, 3429, 35944, 37467, 39277, 41245, 43326, 45649, 48152, 51897, 54686, 57877, 61263, 64962, 68983, 73945, 78619, 83537, 89622, 95002, 1.0067, 1.0742, 1.1355, 1.2007,
01111
01112
01113
01114
                1.2738, 1.347, 1.4254, 1.5094, 1.6009, 1.6976, 1.8019, 1.9148,
01115
                2.0357, 2.166, 2.3066, 2.4579, 2.6208, 2.7966, 2.986, 3.188, 3.4081, 3.6456, 3.9, 4.1747, 4.4712, 4.7931, 5.1359, 5.5097,
01116
01117
                5.9117, 6.3435, 6.8003, 7.3001, 7.8385, 8.3945, 9.011, 9.6869, 10.392, 11.18, 12.036, 12.938, 13.944, 14.881, 16.029, 17.255,
01118
                18.574, 19.945, 21.38, 22.9, 24.477, 26.128, 27.87, 29.037,
                30.988, 33.145, 35.506, 37.76, 40.885, 44.487, 48.505, 52.911, 57.56, 61.964, 67.217, 72.26, 78.343, 85.08, 91.867, 99.435,
01121
01122
                107.68, 116.97, 127.12, 138.32, 150.26, 163.04, 174.81, 189.26, 205.61, 224.68, 240.98, 261.88, 285.1, 307.58, 334.35, 363.53, 394.68, 427.85, 458.85, 489.25, 472.87, 486.93, 496.27, 501.52,
01123
01124
                501.57, 497.14, 488.09, 476.32, 393.76, 388.51, 393.42, 414.45, 455.12, 514.62, 520.38, 547.42, 562.6, 487.47, 480.83, 391.06,
01127
01128
                376.92, 303.7, 295.91, 256.03, 236.73, 280.38, 310.71, 335.53,
               367.88, 401.94, 435.52, 469.13, 497.94, 588.82, 597.94, 597.2, 588.28, 571.2, 555.75, 603.56, 638.15, 680.75, 801.72, 848.01, 962.15, 990.06, 1068.1, 1076.2, 1115.3, 1134.2, 1136.6, 1119.1, 1108.9, 1090.6, 1068.7, 1041.9, 1005.4, 967.98, 927.08, 780.1, 751.41, 733.12, 742.65, 785.56, 855.16, 852.45, 878.1, 784.59, 787.51
01129
01130
01131
01133
01134
                777.81, 765.13, 622.93, 498.09, 474.89, 386.9, 378.48, 336.17,
01135
                322.04, 329.57, 350.5, 383.38, 420.02, 462.39, 499.71, 531.98,
               554.29, 653.43, 639.99, 605.16, 554.16, 504.42, 540.64, 552.33, 679.46, 699.51, 713.91, 832.17, 919.91, 884.96, 907.57, 846.56, 818.56, 768.93, 706.71, 642.17, 575.95, 515.38, 459.07, 409.02, 364.61, 325.46, 291.1, 260.89, 234.39, 211.01, 190.38, 172.11,
01136
01137
01139
                155.91, 141.49, 128.63, 117.13, 106.84, 97.584, 89.262, 81.756,
01140
               74.975, 68.842, 63.28, 58.232, 53.641, 49.46, 45.649, 42.168, 38.991, 36.078, 33.409, 30.96, 28.71, 26.642, 24.737, 22.985, 21.37, 19.882, 18.512, 17.242, 16.073, 14.987, 13.984, 13.05, 12.186, 11.384, 10.637, 9.9436, 9.2988, 8.6991, 8.141, 7.6215,
01141
01142
01143
01144
                7.1378, 6.6872, 6.2671, 5.8754, 5.51, 5.1691, 4.851, 4.5539,
01146
                4.2764, 4.0169, 3.7742, 3.5472, 3.3348, 3.1359, 2.9495, 2.7749,
01147
                2.6113, 2.4578, 2.3139, 2.1789, 2.0523, 1.9334, 1.8219, 1.7171,
01148
                1.6188, 1.5263, 1.4395, 1.3579, 1.2812, 1.209, 1.1411, 1.0773,
               1.0171, .96048, .90713, .85684, .80959, .76495, .72282, .68309, .64563, .61035, .57707, .54573, .51622, .48834, .46199, .43709, .41359, .39129, .37034, .35064, .33198, .31442, .29784, .28218,
01149
01150
                .26732, .25337, .24017, .22774, .21601, .20479, .19426
01152
01153
01154
            static double co2260[2001] = { 5.7971e-5, 6.0733e-5, 6.3628e-5, 6.6662e-5,
01155
               6.9843e-5, 7.3176e-5, 7.6671e-5, 8.0334e-5, 8.4175e-5, 8.8201e-5, 9.2421e-5, 9.6846e-5, 1.0149e-4, 1.0635e-4, 1.1145e-4, 1.1679e-4,
01156
                1.224e-4, 1.2828e-4, 1.3444e-4, 1.409e-4, 1.4768e-4, 1.5479e-4,
01158
01159
                1.6224e-4, 1.7006e-4, 1.7826e-4, 1.8685e-4, 1.9587e-4, 2.0532e-4,
01160
                2.1524e-4, 2.2565e-4, 2.3656e-4, 2.48e-4, 2.6001e-4, 2.7261e-4,
                2.8582e-4, 2.9968e-4, 3.1422e-4, 3.2948e-4, 3.4548e-4, 3.6228e-4,
01161
                3.799e-4, 3.9838e-4, 4.1778e-4, 4.3814e-4, 4.595e-4, 4.8191e-4, 5.0543e-4, 5.3012e-4, 5.5603e-4, 5.8321e-4, 6.1175e-4, 6.417e-4,
01162
                6.7314e-4, 7.0614e-4, 7.4078e-4, 7.7714e-4, 8.1531e-4, 8.5538e-4,
                8.9745e-4, 9.4162e-4, 9.8798e-4, .0010367, .0010878,
01165
                                                                                                       .0011415.
01166
                .0011978, .001257, .0013191, .0013844, .001453, .0015249,
                .0016006, .00168, .0017634, .001851, .001943, .0020397, .0021412, .0022479, .00236, .0024778, .0026015, .0027316, .0028682,
01167
01168
```

```
.0030117, .0031626, .0033211, .0034877, .0036628, .0038469,
                      .0040403, .0042436, .0044574, .004682, .0049182, .0051665, .0054276, .0057021, .0059907, .0062942, .0066133, .0069489,
01170
01171
                       .0073018, .0076729, .0080632, .0084738, .0089056, .0093599,
01172
                      .0098377, .01034, .010869, .011426, .012011, .012627, .013276, .013958, .014676, .015431, .016226, .017063, .017944, .018872, .019848, .020876, .021958, .023098, .024298, .025561, .026892, .028293, .029769, .031323, .032961, .034686, .036503, .038418,
01173
01174
01176
01177
                       .040435, .042561, .044801, .047161, .049649, .052271, .055035,
                      .057948, .061019, .064256, .06767, .07127, .075066, .079069, .083291, .087744, .092441, .097396, .10262, .10814, .11396, .1201, .12658, .13342, .14064, .14826, .1563, .1648, .17376, .18323, .19324, .2038, .21496, .22674, .23919, .25234, .26624, .28093, .29646, .31287, .33021, .34855, .36794, .38844, .41012, .2347, .2348, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .2448, .
01178
01179
01180
01182
01183
                       .43305, .45731, .48297, .51011, .53884, .56924, .60141, .63547,
                      .67152, .70969, .75012, .79292, .83826, .8863, .93718, .99111, 1.0482, 1.1088, 1.173, 1.2411, 1.3133, 1.3898, 1.471, 1.5571, 1.6485, 1.7455, 1.8485, 1.9577, 2.0737, 2.197, 2.3278, 2.4668,
01184
01185
01186
                       2.6145, 2.7715, 2.9383, 3.1156, 3.3042, 3.5047, 3.7181, 3.9451,
                       4.1866, 4.4437, 4.7174, 5.0089, 5.3192, 5.65, 6.0025, 6.3782,
01188
                      6.7787, 7.206, 7.6617, 8.1479, 8.6669, 9.221, 9.8128, 10.445, 11.12, 11.843, 12.615, 13.441, 14.325, 15.271, 16.283, 17.367
01189
01190
01191
                      18.529, 19.776, 21.111, 22.544, 24.082, 25.731, 27.504, 29.409,
                      31.452, 33.654, 36.024, 38.573, 41.323, 44.29, 47.492, 50.951, 54.608, 58.588, 62.929, 67.629, 72.712, 78.226, 84.207, 90.699,
01192
01193
                       97.749, 105.42, 113.77, 122.86, 132.78, 143.61, 155.44, 168.33,
                       182.48, 198.01, 214.87, 233.39, 253.86, 276.34, 300.3, 327.28,
01195
01196
                       356.89, 389.48, 422.29, 458.99, 501.39, 548.13, 595.62, 652.74,
                      716.54, 784.57, 866.78, 960.59, 1062.8, 1072.5, 1189.5, 1319.4, 1467.6, 1630.2, 1813.7, 2016.9, 2253., 2515.3, 2773.5, 3092.8, 3444.4, 3720.4, 4104.3, 4527.5, 4645.9, 5021.7, 5462.2, 5597.,
01197
01198
01199
                       6110.6, 6732.5, 7513.8, 8270.6, 9640.6, 11487., 2796.1, 2680.1,
                       2441.6, 2404.2, 2334.8, 2215.2, 1642.5, 1477.9, 1328.1, 1223.5,
01201
01202
                       843.34, 766.96, 831.65, 834.84, 774.85, 1156.3, 1275.6, 1366.1,
                      1795.6, 1885., 1936.5, 1953.4, 2154.4, 2002.7, 1789.8, 10381., 9040., 8216.5, 7384.7, 6721.9, 6187.7, 6143.8, 5703.9, 5276.6, 4873.1, 4736., 4325.3, 3927., 3554.1, 3286.1, 2950.1, 2642.4, 2368.7, 2138.9, 1914., 1719.6, 1543.9, 1388.6, 1252.1, 1132.2,
01203
01204
01205
01207
                       1024.1, 1025.4, 920.58, 829.59, 750.54, 685.01, 624.25, 570.14,
                       525.81, 481.85, 441.95, 408.71, 377.23, 345.86, 318.51, 292.26,
01208
01209
                      268.34, 247.04, 227.14, 209.02, 192.69, 177.59, 163.78, 151.26,
                      78.13, 72.13, 129.19, 119.53, 110.7, 102.57, 95.109, 88.264, 81.948, 76.13, 70.768, 65.827, 61.251, 57.022, 53.495, 49.824, 46.443, 43.307, 40.405, 37.716, 35.241, 32.923, 30.77, 28.78, 26.915,
01210
01211
                       25.177, 23.56, 22.059, 20.654, 19.345, 18.126, 16.988, 15.93,
01213
01214
                      14.939, 14.014, 13.149, 12.343, 11.589, 10.884, 10.225, 9.6093,
01215
                       9.0327, 8.4934, 7.9889, 7.5166, 7.0744, 6.6604, 6.2727, 5.9098,
                      5.5701, 5.2529, 4.955, 4.676, 4.4148, 4.171, 3.9426, 3.7332, 3.5347, 3.3493, 3.1677, 3.0025, 2.8466, 2.6994, 2.5601, 2.4277, 2.3016, 2.1814, 2.0664, 1.9564, 1.8279, 1.7311, 1.6427, 1.5645,
01216
01217
01218
                      1.4982, 1.443, 1.374, 1.3146, 1.2562, 1.17, 1.1105, 1.0272,
                      .96863, .89718, .83654, .80226, .75908, .72431, .69573, .67174,
01220
01221
                       .65126, .63315, .61693, .60182, .58715, .59554, .57649, .55526,
                      .53177, .50622, .48176, .4813, .47642, .47492, .50273, .50293, .52687, .52239, .53419, .53814, .52626, .52211, .51492, .50622,
01222
01223
                      .49746, .48841, .4792, .43534, .41999, .40349, .38586, .36799, .35108, .31089, .30803, .3171, .33599, .35041, .36149, .32924, .32462, .27309, .25961, .20922, .19504, .15683, .13098, .11588,
01224
01226
                       .11478, .11204, .11363, .12135, .16423, .17785, .19094, .20236,
01227
                     .11476, .11204, .11363, .12135, .16425, .17785, .19094, .20236, .21084, .2154, .24108, .22848, .20871, .18797, .17963, .17834, .21552, .22284, .26945, .27052, .30108, .28977, .29772, .29224, .27658, .24956, .22777, .20654, .18392, .16338, .1452, .12916, .1152, .10304, .092437, .083163, .075031, .067878, .061564, .055976, .051018, .046609, .042679, .03917, .036032, .033223, .030706, .02845, .026428, .024617, .022998, .021554, .02027, .010136, .01841, .013738, .016541, .015026, .016561
01228
01229
01230
01232
01233
01234
                       .019136, .018141, .017278, .016541, .015926, .015432, .015058,
                      .014807, .014666, .014635, .014728, .014947, .01527, .015728, .016345, .017026, .017798, .018839, .019752, .020636, .021886, .022695, .02327, .023478, .024292, .023544, .022222, .021932, .020052, .018143, .017722, .017031, .017782, .01938, .020734,
01235
01236
                       .020476, .019255, .017477, .016878, .014617, .012489, .011765,
01239
01240
                       .0099077, .0086446, .0079446, .0078644, .0079763, .008671,
                      .01001, .0108, .012933, .015349, .016341, .018484, .020254, .020254, .020254, .020478, .019591, .018595, .018385, .019913, .022254,
01241
01242
                      .024847, .025809, .028053, .029924, .030212, .031367, .03222, .032739, .032537, .03286, .033344, .033507, .033499, .033339, .032809, .033041, .031723, .029837, .027511, .026603, .024032,
01243
01244
01245
01246
                       .021914, .020948, .021701, .023425, .024259, .024987, .023818,
01247
                       .021768, .019223, .018144, .015282, .012604, .01163, .0097907,
                      .008336, .0082473, .0079582, .0088077, .009779, .010129, .012145, .014378, .016761, .01726, .018997, .019998, .019809, .01819, .016358, .016099, .01617, .017939, .020223, .022521, .02277,
01248
01249
                      .024279, .025247, .024222, .023989, .023224, .021493, .020362, .018596, .017309, .015975, .014466, .013171, .011921, .01078,
01251
01252
01253
                       .0097229, \ .0087612, \ .0078729, \ .0070682, \ .0063494, \ .0057156,
                      .0051459, .0046273, .0041712, .0037686, .0034119, .003095, .0028126, .0025603, .0023342, .0021314, .0019489, .0017845,
01254
01255
```

```
.001636, .0015017, .00138, .0012697, .0011694, .0010782,
               9.9507e-4, 9.1931e-4, 8.5013e-4, 7.869e-4, 7.2907e-4, 6.7611e-4, 6.2758e-4, 5.8308e-4, 5.4223e-4, 5.0473e-4, 4.7027e-4, 4.3859e-4,
01257
01258
01259
                4.0946e-4, 3.8265e-4, 3.5798e-4, 3.3526e-4, 3.1436e-4, 2.9511e-4,
               2.7739e-4, 2.6109e-4, 2.4609e-4, 2.3229e-4, 2.1961e-4, 2.0797e-4, 1.9729e-4, 1.875e-4, 1.7855e-4, 1.7038e-4, 1.6294e-4, 1.5619e-4,
01260
01261
                1.5007e-4, 1.4456e-4, 1.3961e-4, 1.3521e-4, 1.3131e-4, 1.2789e-4,
                1.2494e-4, 1.2242e-4, 1.2032e-4, 1.1863e-4, 1.1733e-4, 1.1641e-4,
01263
                1.1585e-4, 1.1565e-4, 1.158e-4, 1.1629e-4, 1.1712e-4, 1.1827e-4,
01264
01265
               1.1976e-4, 1.2158e-4, 1.2373e-4, 1.262e-4, 1.2901e-4, 1.3214e-4,
                1.3562e-4, 1.3944e-4, 1.4361e-4, 1.4814e-4, 1.5303e-4, 1.5829e-4,
01266
                1.6394e-4, 1.6999e-4, 1.7644e-4, 1.8332e-4, 1.9063e-4, 1.984e-4,
01267
               2.0663e-4, 2.1536e-4, 2.246e-4, 2.345e-4, 2.4468e-4, 2.5558e-4, 2.6708e-4, 2.7921e-4, 2.92e-4, 3.0548e-4, 3.1968e-4, 3.3464e-4,
01268
01269
01270
               3.5039e-4, 3.6698e-4, 3.8443e-4, 4.0281e-4, 4.2214e-4, 4.4248e-4,
               4.6389e-4, 4.864e-4, 5.1009e-4, 5.3501e-4, 5.6123e-4, 5.888e-4, 6.1781e-4, 6.4833e-4, 6.8043e-4, 7.142e-4, 7.4973e-4, 7.8711e-4, 8.2644e-4, 8.6783e-4, 9.1137e-4, 9.5721e-4, .0010054, .0010562,
01271
01272
01273
               .0011096, .0011659, .0012251, .0012875, .0013532, .0014224, .0014953, .001572, .0016529, .0017381, .0018279, .0019226, .0020224, .0021277, .0022386, .0023557, .0024792, .0026095,
01276
01277
                .002747,\ .0028921,\ .0030453,\ .0032071,\ .003378,\ .0035586,
               .0037494, .003951, .0041642, .0043897, .0046282, .0048805, .0051476, .0054304, .00573, .0060473, .0063837, .0067404,
01278
01279
               .0071188, .0075203, .0079466, .0083994, .0088806, .0093922, .0099366, .010516, .011134, .011792, .012494, .013244, .014046, .014898, .015808, .016781, .017822, .018929, .020108, .02138,
01280
01282
               .014898, .015808, .016781, .017822, .018929, .020108, .02138, .022729, .02419, .02576, .027412, .029233, .031198, .0333301, .035594, .038092, .040767, .04372, .046918, .050246, .053974, .058009, .061976, .066586, .071537, .076209, .081856, .087998, .093821, .10113, .10913, .11731, .12724, .13821, .15025, .1639, .17807, .19472, .21356, .23496, .25758, .28387, .31389, .34104, .37469, .40989, .43309, .46845, .5042, .5023, .52981, .55275,
01283
01284
01285
01286
01288
01289
               .51075, .51976, .52457, .44779, .44721, .4503, .4243, .45244,
01290
                .49491, .55399, .39021, .24802, .2501, .2618, .27475, .28879
               . 31317, . 33643, . 36257, . 4018, . 43275, . 46525, . 53333, . 56599, . 60557, . 70142, . 74194, . 77736, . 88567, . 91182, . 93294, . 98407, . 98772, . 99176, . 9995, 1.2405, 1.3602, 1.338, 1.3255, 1.3267, 1.3404, 1.3634, 1.3967, 1.4407, 1.4961, 1.5603, 1.6328, 1.7153,
01291
01292
01294
                1.8094, 1.9091, 2.018, 2.1367, 2.264, 2.4035, 2.5562, 2.7179,
01295
01296
               2.9017, 3.1052, 3.3304, 3.5731, 3.8488, 4.1553, 4.4769, 4.7818,
               5.1711, 5.5204, 5.9516, 6.4097, 6.8899, 7.1118, 7.5469, 7.9735, 7.9511, 8.3014, 8.6418, 8.4757, 8.8256, 9.2294, 9.6923, 10.033, 10.842, 11.851, 11.78, 8.8435, 9.1381, 9.5956, 10.076, 10.629,
01297
01298
               11.22, 11.883, 12.69, 13.163, 13.974, 14.846, 16.027, 17.053,
01300
01301
                18.148, 19.715, 20.907, 22.163, 23.956, 25.235, 26.566, 27.94
01302
               29.576, 30.956, 32.432, 35.337, 39.911, 41.128, 42.625, 44.386,
               46.369, 48.619, 51.031, 53.674, 56.825, 59.921, 63.286, 66.929, 70.859, 75.081, 79.618, 84.513, 89.739, 95.335, 101.35, 107.76,
01303
01304
               114.63, 121.98, 129.87, 138.3, 147.34, 157.04, 167.56, 178.67,
01305
                190.61, 203.43, 217.19, 231.99, 247.88, 264.98, 283.37, 303.17,
                324.49, 347.47, 372.25, 398.98, 427.85, 459.06, 492.8, 529.31,
01307
01308
                568.89, 611.79, 658.35, 708.91, 763.87, 823.65, 888.72, 959.58
               1036.8, 1121.8, 1213.9, 1314.3, 1423.8, 1543., 1672.8, 1813.4, 1966.1, 2131.4, 2309.5, 2499.3, 2705., 2925.7, 3161.6, 3411.3, 3611.5, 3889.2, 4191.1, 4519.3, 4877.9, 5272.9, 5712.9, 6142.7 6719.6, 7385., 8145., 8977.7, 9831.9, 10827., 11934., 13063.,
01309
01310
01311
                14434., 15878., 17591., 19435., 21510., 23835., 26835., 29740.,
01313
                32878., 36305., 39830., 43273., 46931., 50499., 49586., 51598.,
01314
01315
               53429., 54619., 55081., 55102., 54485., 53487., 52042., 42689.,
                42607., 44020., 47994., 54169., 53916., 55808., 56642., 46049.,
01316
               44243., 32929., 30658., 21963., 20835., 15962., 13679., 17652., 19680., 22388., 25625., 29184., 32520., 35720., 38414., 40523.,
01317
                49228., 48173., 45678., 41768., 37600., 41313., 42654., 44465.,
                55736., 56630., 65409., 63308., 66572., 61845., 60379., 56777.,
01320
01321
               51920., 46601., 41367., 36529., 32219., 28470., 25192., 22362.
               19907., 17772., 15907., 14273., 12835., 11567., 10445., 9450.2, 8565.1, 7776., 7070.8, 6439.2, 5872.3, 5362.4, 4903., 4488.3, 4113.4, 3773.8, 3465.8, 3186.1, 2931.7, 2700.1, 2488.8, 2296., 2119.8, 1958.6, 1810.9, 1675.6, 1551.4, 1437.3, 1332.4, 1236.,
01322
01323
01324
               1147.2, 1065.3, 989.86, 920.22, 855.91, 796.48, 741.53, 690.69, 643.62, 600.02, 559.6, 522.13, 487.35, 455.06, 425.08, 397.21,
01326
01327
               371.3, 347.2, 324.78, 303.9, 284.46, 266.34, 249.45, 233.7, 219.01, 205.3, 192.5, 180.55, 169.38, 158.95, 149.2, 140.07,
01328
01329
               131.54, 123.56, 116.09, 109.09, 102.54, 96.405, 90.655, 85.266, 80.213, 75.475, 71.031, 66.861, 62.948, 59.275, 55.827, 52.587,
01330
01331
                49.544, 46.686, 43.998, 41.473, 39.099, 36.867, 34.768, 32.795,
01332
01333
                30.939, 29.192, 27.546, 25.998, 24.539, 23.164, 21.869, 20.65,
               19.501, 18.419, 17.399, 16.438, 15.532, 14.678, 13.874, 13.115,
01334
               12.4, 11.726, 11.088, 10.488, 9.921, 9.3846, 8.8784, 8.3996, 7.9469, 7.5197, 7.1174, 6.738, 6.379, 6.0409, 5.7213, 5.419, 5.1327, 4.8611, 4.6046, 4.3617, 4.1316, 3.9138, 3.7077, 3.5125,
01335
01336
                3.3281, 3.1536, 2.9885, 2.8323, 2.6846, 2.5447, 2.4124, 2.2871,
01338
               2.1686, 2.0564, 1.9501, 1.8495, 1.7543, 1.6641, 1.5787, 1.4978,
01339
01340
               1.4212, 1.3486, 1.2799, 1.2147, 1.1529, 1.0943, 1.0388, .98602,
               .93596, .8886, .84352, .80078, .76029, .722, .68585, .65161, .61901, .58808, .55854, .53044, .5039, .47853, .45459, .43173,
01341
01342
```

```
01343
                     .41008, .38965, .37021, .35186, .33444, .31797, .30234, .28758,
                     .2736, .26036, .24764, .2357, .22431, .21342, .20295, .19288, .18334, .17444, .166, .15815, .15072, .14348, .13674, .13015,
01344
01345
                     12399, 11807, 11231, 10689, 10164, 096696, 091955, 087476, 083183, 079113, 075229, 071536, 068026, 064698, 06154, 058544, 055699, 052997, 050431, 047993, 045676, 043475, 041382, 039392, 037501, 035702, 033991, 032364, 030817, 029345, 027945, 026613, 025345, 024139, 022991,
01346
01347
01348
01350
                    .03081/, .029345, .027945, .026615, .025345, .024139, .022991, .021899, .02086, .019871, .018929, .018033, .01718, .016368, .015595, .014859, .014158, .013491, .012856, .012251, .011675, .011126, .010604, .010107, .0096331, .009182, .0087523, .0083431, .0079533, .0075821, .0072284, .0068915, .0065706, .0062649, .0059737, .0056963, .005432, .0051802, .0049404, .0047118, .0044941, .0042867, .0040891, .0039009, .0037216, .0035507, .003388, .0032329, .0030852, .0029445, .0028105, .0026829, .0025613, .0024455, .0023353, .0023304, .0020353
01351
01352
01353
01354
01356
01357
                     .0025613, .0024455, .0023353, .0022303, .0021304, .0020353, .0019448, .0018587, .0017767, .0016988, .0016247, .0015543,
01358
01359
                     .0014874, .0014238, .0013635, .0013062, .0012519, .0012005, .0011517, .0011057, .0010621, .001021, 9.8233e-4, 9.4589e-4,
01360
                      9.1167e-4, 8.7961e-4, 8.4964e-4, 8.2173e-4, 7.9582e-4, 7.7189e-4,
01362
                     7.499e-4, 7.2983e-4, 7.1167e-4, 6.9542e-4, 6.8108e-4, 6.6866e-4,
01363
01364
                     6.5819e-4, 6.4971e-4, 6.4328e-4, 6.3895e-4, 6.3681e-4, 6.3697e-4,
                    6.3956e-4, 6.4472e-4, 6.5266e-4, 6.6359e-4, 6.778e-4, 6.9563e-4, 7.1749e-4, 7.4392e-4, 7.7556e-4, 8.1028e-4, 8.4994e-4, 8.8709e-4, 9.3413e-4, 9.6953e-4, .0010202, .0010738, .0010976, .0011507, .0011686, .0012264, .001291, .0013346, .0014246, .0015293, .0016359, .0017824, .0019255, .0020854, .002247, .0024148,
01365
01366
01367
01368
01369
01370
                      .0026199, .0027523, .0029704, .0030702, .0033047, .0035013,
01371
                     .0037576, .0040275, .0043089, .0046927, .0049307, .0053486,
01372
                     .0053809, \ .0056699, \ .0059325, \ .0055488, \ .005634, \ .0056392,
                     .004946, .0048855, .0048208, .0044386, .0045498, .0046377, .0048939, .0052396, .0057324, .0060859, .0066906, .0071148,
01373
                     .0077224, .0082687, .008769, .0084471, .008572, .0087729, .008775, .0090742, .0080704, .0080288, .0085747, .0086087,
01375
01376
                     .0086408, .0088752, .0089381, .0089757, .0093532, .0092824, .0092566, .0092645, .0092735, .009342, .0095806, .0097991,
01377
01378
                     .010213, .010611, .011129, .011756, .013237, .01412, .015034, .015936, .01682, .018597, .019315, .019995, .020658, .021289, .022363, .022996, .023716, .024512, .025434, .026067, .027118,
01379
01381
                     .028396, .029865, .031442, .033253, .03525, .037296, .039701, .042356, .045154, .048059, .051294, .054893, .058636, .061407
01382
01383
                     .065172, .068974, .072676, .073379, .076547, .079556, .079134, .082308, .085739, .090192, .09359, .099599, .10669, .11496,
01384
01385
                     .1244, .13512, .14752, .14494, .15647, .1668, .17863, .19029, .20124, .20254, .21179, .21982, .21625, .22364, .23405, .23382, .2434, .25708, .26406, .27621, .28909, .30395, .31717, .33271,
01386
01387
01388
01389
                      .3496, .36765, .38774, .40949, .446, .46985, .49846, .5287, .562,
                     59841, .64598, .68834, .7327, .78978, .8373, .88708, .94744, 1.0006, 1.0574, 1.1215, 1.1856, 1.2546, 1.3292, 1.4107, 1.4974, 1.5913, 1.6931, 1.8028, 1.9212, 2.0492, 2.1874, 2.3365, 2.4978, 2.6718, 2.8588, 3.062, 3.2818, 3.5188, 3.7752, 4.0527, 4.3542, 4.6782, 5.0312, 5.4123, 5.8246, 6.2639, 6.7435, 7.2636, 7.8064,
01390
01391
01392
01394
01395
                     8.4091, 9.0696, 9.7677, 10.548, 11.4, 12.309, 13.324, 14.284,
                     15.445, 16.687, 18.019, 19.403, 20.847, 22.366, 23.925, 25.537, 27.213, 28.069, 29.864, 31.829, 33.988, 35.856, 38.829, 42.321, 46.319, 50.606, 55.126, 59.126, 64.162, 68.708, 74.615, 81.176, 87.739, 95.494, 103.83, 113.38, 123.99, 135.8, 148.7, 162.58,
01396
01397
01398
                     176.32, 192.6, 211.47, 232.7, 252.64, 277.41, 305.38, 333.44,
01400
                     366.42, 402.66, 442.14, 484.53, 526.42, 568.15, 558.78, 582.6, 600.98, 613.94, 619.44, 618.24, 609.84, 595.96, 484.86, 475.59,
01401
01402
                     478.49, 501.56, 552.19, 628.44, 630.39, 658.92, 671.96, 562.7, 545.88, 423.43, 400.14, 306.59, 294.13, 246.8, 226.51, 278.21, 314.39, 347.22, 389.13, 433.16, 477.48, 521.67, 560.54, 683.6, 696.37, 695.91, 683.1, 658.24, 634.89, 698.85, 742.87, 796.66, 954.49, 1009.5, 1150.5, 1179.1, 1267.9, 1272.4, 1312.7, 1330.4,
01403
01404
01406
01407
01408
                     1331.6, 1315.8, 1308.3, 1293.3, 1274.6, 1249.5, 1213.2, 1172.1,
01409
                     1124.4, 930.33, 893.36, 871.27, 883.54, 940.76, 1036., 1025.6, 1053.1, 914.51, 894.15, 865.03, 670.63, 508.41, 475.15, 370.85,
01410
                    361.06, 319.38, 312.75, 331.87, 367.13, 415., 467.94, 525.49, 578.41, 624.66, 794.82, 796.97, 780.29, 736.49, 670.18, 603.75, 659.67, 679.8, 857.12, 884.05, 900.65, 1046.1, 1141.9, 1083., 1089.2, 1e3, 947.08, 872.31, 787.91, 704.75, 624.93, 553.68,
01411
01413
01414
                     489.91, 434.21, 385.64, 343.3, 306.42, 274.18, 245.94, 221.11,
01415
                     199.23, 179.88, 162.73, 147.48, 133.88, 121.73, 110.86, 101.1, 92.323, 84.417, 77.281, 70.831, 64.991, 59.694, 54.884, 50.509, 46.526, 42.893, 39.58, 36.549, 33.776, 31.236, 28.907, 26.77,
01416
01417
                     24.805, 23., 21.339, 19.81, 18.404, 17.105, 15.909, 14.801,
01419
                     13.778, 12.83, 11.954, 11.142, 10.389, 9.691, 9.0434, 8.4423, 7.8842, 7.3657, 6.8838, 6.4357, 6.0189, 5.6308, 5.2696, 4.9332,
01420
01421
                     4.6198, 4.3277, 4.0553, 3.8012, 3.5639, 3.3424, 3.1355, 2.9422, 2.7614, 2.5924, 2.4343, 2.2864, 2.148, 2.0184, 1.8971, 1.7835, 1.677, 1.5773, 1.4838, 1.3961, 1.3139, 1.2369, 1.1645, 1.0966,
01422
01423
                     1.0329, .97309, .91686, .86406, .81439, .76767, .72381, .68252, .64359, .60695, .57247, .54008, .50957, .48092, .45401, .42862,
01425
01426
                     .40465, .38202, .36072, .34052, .3216, .30386, .28711, .27135, .25651, .24252, .2293, .21689, .20517, .19416, .18381, .17396,
01427
01428
                      .16469
```

```
01430
01431
01432
             static double co2230[2001] = {2.743e-5, 2.8815e-5, 3.027e-5, 3.1798e-5,}
01433
                3.3405e-5, 3.5094e-5, 3.6869e-5, 3.8734e-5, 4.0694e-5, 4.2754e-5,
                4.492e-5, 4.7196e-5, 4.9588e-5, 5.2103e-5, 5.4747e-5, 5.7525e-5, 6.0446e-5, 6.3516e-5, 6.6744e-5, 7.0137e-5, 7.3704e-5, 7.7455e-5,
01434
01435
                8.1397e-5, 8.5543e-5, 8.9901e-5, 9.4484e-5, 9.9302e-5, 1.0437e-4,
                1.097e-4, 1.153e-4, 1.2119e-4, 1.2738e-4, 1.3389e-4, 1.4074e-4,
01438
                1.4795e-4, 1.5552e-4, 1.6349e-4, 1.7187e-4, 1.8068e-4, 1.8995e-4,
                1.997e-4, 2.0996e-4, 2.2075e-4, 2.321e-4, 2.4403e-4, 2.5659e-4, 2.698e-4, 2.837e-4, 2.9832e-4, 3.137e-4, 3.2988e-4, 3.4691e-4,
01439
01440
                3.6483e-4, 3.8368e-4, 4.0351e-4, 4.2439e-4, 4.4635e-4, 4.6947e-4,
01441
                4.9379e-4, 5.1939e-4, 5.4633e-4, 5.7468e-4, 6.0452e-4,
01442
                6.69e-4, 7.038e-4, 7.4043e-4, 7.79e-4, 8.1959e-4, 8.6233e-4,
01443
01444
                9.0732e-4, 9.5469e-4, .0010046, .0010571, .0011124, .0011706
01445
                .0012319, .0012964, .0013644, .001436, .0015114, .0015908,
                .0016745, .0017625, .0018553, .0019531, .002056, .0021645, .0022788, .0023992, .002526, .0026596, .0028004, .0029488, .0031052, .0032699, .0034436, .0036265, .0038194, .0040227, .0042369, .0044628, .0047008, .0049518, .0052164, .0054953,
01446
01447
                .0057894, .0060995, .0064265, .0067713, .007135, .0075184, .0079228, .0083494, .0087993, .0092738, .0097745, .010303,
01450
01451
                .01086, .011448, .012068, .012722, .013413, .014142, .014911, .015723, .01658, .017484, .018439, .019447, .020511, .021635, .022821, .024074, .025397, .026794, .02827, .029829, .031475, .033215, .035052, .036994, .039045, .041213, .043504, .045926, .048485, .05119, .05405, .057074, .060271, .063651, .067225,
01452
01453
01454
01456
01457
                .071006, .075004, .079233, .083708, .088441, .093449, .098749,
                .07304, .07304, .079233, .003706, .003441, .093449, .093749, .10436, .11029, .11657, .12322, .13026, .13772, .14561, .15397, .16282, .1722, .18214, .19266, .20381, .21563, .22816, .24143, .2555, .27043, .28625, .30303, .32082, .3397, .35972, .38097, .40352, .42746, .45286, .47983, .50847, .53888, .57119, .6055, .64196, .6807, .72187, .76564, .81217, .86165, .91427, .97025,
01458
01459
01460
01462
01463
                1.0298, 1.0932, 1.1606, 1.2324, 1.3088, 1.3902, 1.477, 1.5693,
                1.6678, 1.7727, 1.8845, 2.0038, 2.131, 2.2666, 2.4114, 2.5659, 2.7309, 2.907, 3.0951, 3.2961, 3.5109, 3.7405, 3.986, 4.2485, 4.5293, 4.8299, 5.1516, 5.4961, 5.8651, 6.2605, 6.6842, 7.1385, 7.6256, 8.1481, 8.7089, 9.3109, 9.9573, 10.652, 11.398, 12.2, 13.063, 13.992, 14.99, 16.064, 17.222, 18.469, 19.813, 21.263,
01464
01465
01466
                22.828, 24.516, 26.34, 28.31, 30.437, 32.738, 35.226, 37.914,
01469
01470
                40.824, 43.974, 47.377, 51.061, 55.011, 59.299, 63.961, 69.013,
                74.492, 80.444, 86.919, 93.836, 101.23, 109.25, 117.98, 127.47,
01471
                137.81, 149.07, 161.35, 174.75, 189.42, 205.49, 223.02, 242.26, 263.45, 286.75, 311.94, 340.01, 370.86, 404.92, 440.44, 480.27, 525.17, 574.71, 626.22, 686.8, 754.38, 827.07, 913.38, 1011.7,
01472
01473
01474
01475
                1121.5, 1161.6, 1289.5, 1432.2, 1595.4, 1777., 1983.3, 2216.1,
01476
                2485.7, 2788.3, 3101.5, 3481., 3902.1, 4257.1, 4740., 5272.8,
                5457.9, 5946.2, 6505.3, 6668.4, 7302.4, 8061.6, 9015.8, 9908.3, 11613., 13956., 3249.6, 3243., 2901.5, 2841.3, 2729.6, 2558.2, 1797.8, 1583.2, 1386., 1233.5, 787.74, 701.46, 761.66, 767.21,
01477
01478
01479
                 722.83, 1180.6, 1332.1, 1461.6, 2032.9, 2166., 2255.9, 2294.7,
                2587.2, 2396.5, 2122.4, 12553., 10784., 9832.5, 8827.3, 8029.1
01481
01482
                7377.9, 7347.1, 6783.8, 6239.1, 5721.1, 5503., 4975.1, 4477.8,
01483
                4021.3, 3676.8, 3275.3, 2914.9, 2597.4, 2328.2, 2075.4, 1857.6,
                1663.6, 1493.3, 1343.8, 1213.3, 1095.6, 1066.5, 958.91, 865.15,
01484
                783.31, 714.35, 650.77, 593.98, 546.2, 499.9, 457.87, 421.75, 387.61, 355.25, 326.62, 299.7, 275.21, 253.17, 232.83, 214.31, 197.5, 182.08, 167.98, 155.12, 143.32, 132.5, 122.58, 113.48,
01485
01487
                105.11, 97.415, 90.182, 83.463, 77.281, 71.587, 66.341, 61.493,
01488
                57.014, 53.062, 49.21, 45.663, 42.38, 39.348, 36.547, 33.967,
01489
                31.573, 29.357, 27.314, 25.415, 23.658, 22.03, 20.524, 19.125,
01490
                17.829, 16.627, 15.511, 14.476, 13.514, 12.618, 11.786, 11.013, 10.294, 9.6246, 9.0018, 8.4218, 7.8816, 7.3783, 6.9092, 6.4719,
01491
01492
                6.0641, 5.6838, 5.3289, 4.998, 4.6893, 4.4014, 4.1325, 3.8813, 3.6469, 3.4283, 3.2241, 3.035, 2.8576, 2.6922, 2.5348, 2.3896,
01494
01495
                2.2535, 2.1258, 2.0059, 1.8929, 1.7862, 1.6854, 1.5898, 1.4992,
                1.4017, 1.3218, 1.2479, 1.1809, 1.1215, 1.0693, 1.0116, .96016, .9105, .84859, .80105, .74381, .69982, .65127, .60899, .57843, .54592, .51792, .49336, .47155, .45201, .43426, .41807, .40303,
01496
01497
01498
                .38876, .3863, .37098, .35492, .33801, .32032, .30341, .29874, .29193, .28689, .29584, .29155, .29826, .29195, .29287, .2904, .28199, .27709, .27162, .26622, .26133, .25676, .25235, .23137
01500
01501
01502
                 .22365, .21519, .20597, .19636, .18699, .16485, .16262, .16643,
                .17542, .18198, .18631, .16759, .16338, .13505, .1267, .10053,
01503
                .092554, .074093, .062159, .055523, .054849, .05401, .05528, .058982, .07952, .08647, .093244, .099285, .10393, .10661,
01504
                .12072, .11417, .10396, .093265, .089137, .088909,
01506
                                                                                                       .10902,
                .11277, .13625, .13565, .14907, .14167, .1428, .13744, .12768, .11382, .10244, .091686, .08109, .071739, .063616, .056579, .050504, .045251, .040689, .036715, .033237, .030181, .027488,
01507
01508
01509
                .025107, .022998, .021125, .01946, .017979, .016661, .015489, .014448, .013526, .012712, .011998, .011375, .010839, .010384
01510
                .010007, .0097053, .0094783, .0093257, .0092489, .0092504, .0093346, .0095077, .0097676, .01012, .01058, .011157, .011844,
01512
01513
01514
                 .012672, .013665, .014766, .015999, .017509, .018972, .020444,
                .022311, .023742, .0249, .025599, .026981, .026462, .025143, .025066, .022814, .020458, .020026, .019142, .020189, .022371,
01515
01516
```

```
.024163, .023728, .02199, .019506, .018591, .015576, .012784,
                    .011744, .0094777, .0079148, .0070652, .006986, .0071758, .008086, .0098025, .01087, .013609, .016764, .018137, .021061,
01518
01519
01520
                     .023498, .023576, .023965, .022828, .021519, .021283, .023364,
                    .026457, .029782, .030856, .033486, .035515, .035543, .036558, .037198, .037472, .037045, .037284, .03777, .038085, .038366, .038526, .038282, .038915, .037697, .035667, .032941, .031959, .028692, .025918, .024596, .025592, .027873, .028935, .02984,
01521
01524
01525
                     .028148, .025305, .021912, .020454, .016732, .013357, .01205,
                    .028148, .025305, .021912, .020454, .016/32, .01335/, .01205, .009731, .0079881, .0077704, .0074387, .0083895, .0096776, .010326, .01293, .015955, .019247, .020145, .02267, .024231, .024184, .022131, .019784, .01955, .01971, .022119, .025116, .027978, .028107, .029808, .030701, .029164, .028551, .027286, .024946, .023259, .020982, .019221, .017471, .015643, .014074, .01261, .011301, .010116, .0090582, .0081036, .0072542, .0065034, .0052426, .0052571, .0047321, .004787, .0038677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0034677, .0
01526
01527
01528
01530
01531
                     .0058436, .0052571, .0047321, .0042697, .0038607, .0034977, .0031747, .0028864, .0026284, .002397, .002189, .0020017,
01532
01533
                     .0018326, .0016798, .0015414, .0014159, .0013019, .0011983, .0011039, .0010177, 9.391e-4, 8.6717e-4, 8.0131e-4, 7.4093e-4,
01534
                     6.8553e-4, 6.3464e-4, 5.8787e-4, 5.4487e-4, 5.0533e-4, 4.69e-4,
01536
                     4.3556e-4, 4.0474e-4, 3.7629e-4, 3.5e-4, 3.2569e-4, 3.032e-4,
01537
01538
                     2.8239e-4, 2.6314e-4, 2.4535e-4, 2.2891e-4, 2.1374e-4, 1.9975e-4,
                     1.8685e-4, 1.7498e-4, 1.6406e-4, 1.5401e-4, 1.4479e-4, 1.3633e-4,
01539
                     1.2858e-4, 1.2148e-4, 1.1499e-4, 1.0907e-4, 1.0369e-4, 9.8791e-5,
01540
                     9.4359e-5, 9.0359e-5, 8.6766e-5, 8.3555e-5, 8.0703e-5, 7.8192e-5,
                     7.6003e-5, 7.4119e-5, 7.2528e-5, 7.1216e-5, 7.0171e-5, 6.9385e-5,
                     6.8848e-5, 6.8554e-5, 6.8496e-5, 6.8669e-5, 6.9069e-5, 6.9694e-5,
01543
01544
                     7.054e-5, 7.1608e-5, 7.2896e-5, 7.4406e-5, 7.6139e-5, 7.8097e-5,
                     8.0283e-5, 8.2702e-5, 8.5357e-5, 8.8255e-5, 9.1402e-5, 9.4806e-5, 9.8473e-5, 1.0241e-4, 1.0664e-4, 1.1115e-4, 1.1598e-4, 1.2112e-4,
01545
01546
01547
                     1.2659e-4, 1.3241e-4, 1.3859e-4, 1.4515e-4, 1.521e-4, 1.5947e-4,
                     1.6728e-4, 1.7555e-4, 1.8429e-4, 1.9355e-4, 2.0334e-4, 2.1369e-4,
                     2.2463e-4, 2.3619e-4, 2.4841e-4, 2.6132e-4, 2.7497e-4, 2.8938e-4,
01549
01550
                     3.0462e-4, 3.2071e-4, 3.3771e-4, 3.5567e-4, 3.7465e-4, 3.947e-4,
                     4.1588e-4, 4.3828e-4, 4.6194e-4, 4.8695e-4, 5.1338e-4, 5.4133e-4, 5.7087e-4, 6.0211e-4, 6.3515e-4, 6.701e-4, 7.0706e-4, 7.4617e-4, 7.8756e-4, 8.3136e-4, 8.7772e-4, 9.2681e-4, 9.788e-4, .0010339,
01551
01552
01553
                    .0010922, .001154, .0012195, .0012889, .0013626, .0014407, .0015235, .0016114, .0017048, .0018038, .001909, .0020207,
01555
01556
                     .0021395, .0022657, .0023998, .0025426, .0026944, .002856,
01557
                     .0030281, \ .0032114, \ .0034068, \ .003615, \ .0038371, \ .004074,
                     .004327, .0045971, .0048857, .0051942, .0055239, .0058766, .0062538, .0066573, .0070891, .007551, .0080455, .0085747, .0091412, .0097481, .010397, .011092, .011837, .012638, .013495,
01558
01559
01560
                    .014415, .01541, .016475, .017621, .018857, .020175, .02162, .023185, .024876, .02672, .028732, .030916, .033319, .035939
01561
01562
01563
                     .038736, .041847, .04524, .048715, .052678, .056977, .061203,
01564
                     .066184, .07164, .076952, .083477, .090674, .098049, .10697,
                     . 1169, 1277, 14011, 15323, 1684, 18601, 20626, 22831, 25417, 28407, .31405, .34957, .38823, .41923, .46026, .50409, .51227, .54805, .57976, .53818, .55056, .557, .46741, .46403,
01565
01566
                     .4636, .42265, .45166, .49852, .56663, .34306, .17779, .17697
01568
01569
                     .18346, .19129, .20014, .21778, .23604, .25649, .28676, .31238,
                     .33856, .39998, .4288, .46568, .56654, .60786, .64473, .76466, .7897, .80778, .86443, .85736, .84798, .84157, 1.1385, 1.2446, 1.1923, 1.1552, 1.1338, 1.1266, 1.1292, 1.1431, 1.1683, 1.2059,
01570
01571
01572
                     1.2521, 1.3069, 1.3712, 1.4471, 1.5275, 1.6165, 1.7145, 1.8189, 1.9359, 2.065, 2.2007, 2.3591, 2.5362, 2.7346, 2.9515, 3.2021,
01574
                    1.5339, 2.063, 2.2007, 2.3391, 2.3392, 2.7346, 2.3313, 3.2021, 3.4851, 3.7935, 4.0694, 4.4463, 4.807, 5.2443, 5.7178, 6.2231, 6.4796, 6.9461, 7.4099, 7.3652, 7.7182, 8.048, 7.7373, 8.0363, 8.3855, 8.8044, 9.0257, 9.8574, 10.948, 10.563, 6.8979, 7.0744, 7.4121, 7.7663, 8.1768, 8.6243, 9.1437, 9.7847, 10.182, 10.849, 11.572, 12.602, 13.482, 14.431, 15.907, 16.983, 18.11, 19.884,
01575
01576
01577
01578
                     21.02, 22.18, 23.355, 24.848, 25.954, 27.13, 30.186, 34.893,
01580
01581
                     35.682, 36.755, 38.111, 39.703, 41.58, 43.606, 45.868, 48.573,
01582
                     51.298, 54.291, 57.559, 61.116, 64.964, 69.124, 73.628, 78.471,
                    83.683, 89.307, 95.341, 101.84, 108.83, 116.36, 124.46, 133.18, 142.57, 152.79, 163.69, 175.43, 188.11, 201.79, 216.55, 232.51, 249.74, 268.38, 288.54, 310.35, 333.97, 359.55, 387.26, 417.3, 449.88, 485.2, 523.54, 565.14, 610.28, 659.31, 712.56, 770.43,
01583
01584
01585
                     833.36, 901.82, 976.36, 1057.6, 1146.8, 1243.8, 1350., 1466.3,
01587
01588
                     1593.6, 1732.7, 1884.1, 2049.1, 2228.2, 2421.9, 2629.4, 2853.7,
                     3094.4, 3351.1, 3622.3, 3829.8, 4123.1, 4438.3, 4777.2, 5144.1, 5545.4, 5990.5, 6404.5, 6996.8, 7687.6, 8482.9, 9349.4, 10203.,
01589
01590
                     11223., 12358., 13493., 14916., 16416., 18236., 20222., 22501., 25102., 28358., 31707., 35404., 39538., 43911., 48391., 53193.,
01591
01592
                     58028., 58082., 61276., 64193., 66294., 67480., 67921., 67423.,
01593
01594
                     66254., 64341., 51737., 51420., 53072., 58145., 66195., 65358.,
                     67377., 67869., 53509., 50553., 35737., 32425., 21704., 19974.,
01595
                     14457., 12142., 16798., 19489., 23049., 27270., 31910., 36457.,
01596
                     40877., 44748., 47876., 59793., 58626., 55454., 50337., 44893., 50228., 52216., 54747., 69541., 70455., 81014., 77694., 80533.,
01597
                     73953., 70927., 65539., 59002., 52281., 45953., 40292., 35360., 31124., 27478., 24346., 21647., 19308., 17271., 15491., 13927.,
01599
01600
                     12550., 11331., 10250., 9288.8, 8431.4, 7664.9, 6978.3, 6361.8, 5807.4, 5307.7, 4856.8, 4449., 4079.8, 3744.9, 3440.8, 3164.2, 2912.3, 2682.7, 2473., 2281.4, 2106., 1945.3, 1797.9, 1662.5,
01601
01602
01603
```

```
1538.1, 1423.6, 1318.1, 1221., 1131.5, 1049., 972.99, 902.87,
                838.01, 777.95, 722.2, 670.44, 622.35, 577.68, 536.21, 497.76, 462.12, 429.13, 398.61, 370.39, 344.29, 320.16, 297.85, 277.2,
01606
01607
                258.08, 240.38, 223.97, 208.77, 194.66, 181.58, 169.43, 158.15,
01608
                147.67, 137.92, 128.86, 120.44, 112.6, 105.3, 98.499, 92.166, 86.264, 80.763, 75.632, 70.846, 66.381, 62.213, 58.321, 54.685,
01609
                51.288, 48.114, 45.145, 42.368, 39.772, 37.341, 35.065, 32.937, 30.943, 29.077, 27.33, 25.693, 24.158, 22.717, 21.367, 20.099,
01610
                18.909, 17.792, 16.744, 15.761, 14.838, 13.971, 13.157, 12.393, 11.676, 11.003, 10.369, 9.775, 9.2165, 8.6902, 8.1963, 7.7314, 7.2923, 6.8794, 6.4898, 6.122, 5.7764, 5.4525, 5.1484, 4.8611,
01612
01613
01614
                4.5918, 4.3379, 4.0982, 3.8716, 3.6567, 3.4545, 3.2634, 3.0828,
01615
                2.9122, 2.7512, 2.5993, 2.4561, 2.3211, 2.1938, 2.0737, 1.9603,
                1.8534, 1.7525, 1.6572, 1.5673, 1.4824, 1.4022, 1.3265, 1.2551,
01617
01618
                1.1876, 1.1239, 1.0637, 1.0069, .9532, .90248, .85454, .80921,
               .76631, .72569, .6872, .65072, .61635, .5836, .55261, .52336, .49581, .46998, .44559, .42236, .40036, .37929, .35924, .34043,
01619
01620
               01621
01622
01624
01625
                .04435, .042044, .039866, .037808, .035863, .034023, .032282,
01626
               .030634, .029073, .027595, .026194, .024666, .023608, .022415, .021283, .02021, .019193, .018228, .017312, .016443, .015619, .014837, .014094, .01339, .012721, .012086, .011483, .010911, .010368, .009852, .0093623, .0088972, .0084556, .0080362,
01627
01628
01629
01630
                .0076379, .0072596, .0069003, .006559, .0062349, .0059269, .0056344, .0053565, .0050925, .0048417, .0046034, .004377,
01631
01632
                .0041618, .0039575, .0037633, .0035788, .0034034, .0032368,
01633
                .0030785, .002928, .0027851, .0026492, .0025201, .0023975, .0022809, .0021701, .0020649, .0019649, .0018699, .0017796, .0016938, .0016122, .0015348, .0014612, .0013913, .001325,
01634
01636
01637
                 .0012619, .0012021, .0011452, .0010913, .0010401, 9.9149e-4
01638
                9.454e-4, 9.0169e-4, 8.6024e-4, 8.2097e-4, 7.8377e-4, 7.4854e-4,
               7.1522e-4, 6.8371e-4, 6.5393e-4, 6.2582e-4, 5.9932e-4, 5.7435e-4, 5.5087e-4, 5.2882e-4, 5.0814e-4, 4.8881e-4, 4.7076e-4, 4.5398e-4,
01639
01640
                4.3843e-4, 4.2407e-4, 4.109e-4, 3.9888e-4, 3.88e-4, 3.7826e-4,
01641
                3.6963e-4, 3.6213e-4, 3.5575e-4, 3.505e-4, 3.464e-4, 3.4346e-4,
                3.4173e-4, 3.4125e-4, 3.4206e-4, 3.4424e-4, 3.4787e-4, 3.5303e-4,
01643
01644
                3.5986e-4, 3.6847e-4, 3.7903e-4, 3.9174e-4, 4.0681e-4, 4.2455e-4,
               4.4527e-4, 4.6942e-4, 4.9637e-4, 5.2698e-4, 5.5808e-4, 5.9514e-4, 6.2757e-4, 6.689e-4, 7.1298e-4, 7.3955e-4, 7.8403e-4, 8.0449e-4, 8.5131e-4, 9.0256e-4, 9.3692e-4, .0010051, .0010846, .0011678,
01645
01646
01647
               .001282, .0014016, .0015355, .0016764, .0018272, .0020055, .0021455, .0023421, .0024615, .0026786, .0028787, .0031259,
01649
01650
                .0034046, .0036985, .0040917, .0043902, .0048349, .0049531,
               .0053044, .0056148, .0052452, .0053357, .005333, .0045069, .0043851, .004253, .003738, .0038084, .0039013, .0041505, .0045372, .0050569, .0054507, .0061267, .0066122, .0072449, .0078012, .0082651, .0076538, .0076573, .0076806, .0075227,
01651
01652
01653
                .0076269, .0063758, .006254, .0067749, .0067909, .0068231,
01655
01656
                .0072143, .0072762, .0072954, .007679, .0075107, .0073658,
               .0072441, .0071074, .0070378, .007176, .0072472, .0075844, .0079291, .008412, .0090165, .010688, .011535, .012375, .013166, .013895, .015567, .016011, .016392, .016737, .017043, .017731, .018031, .018419, .018877, .019474, .019868, .020604, .021538, .022653, .023869, .025288, .026879, .028547, .030524, .03274,
01657
01658
01659
01661
                .035132, .03769, .040567, .043793, .047188, .049962, .053542, .057205, .060776, .061489, .064419, .067124, .065945, .068487
01662
01663
               .071209, .074783, .077039, .082444, .08902, .09692, .10617, .11687, .12952, .12362, .13498, .14412, .15492, .16519, .1744, .17096, .17714, .18208, .17363, .17813, .18564, .18295, .19045,
01664
01665
                .20252, .20815, .21844, .22929, .24229, .25321, .26588, .2797, .29465, .31136, .32961, .36529, .38486, .41027, .43694, .4667,
01668
01669
                .49943, .54542, .58348, .62303, .67633, .71755, .76054, .81371,
                .85934, .90841, .96438, 1.0207, 1.0821, 1.1491, 1.2226, 1.3018, 1.388, 1.4818, 1.5835, 1.6939, 1.8137, 1.9435, 2.0843, 2.237, 2.4026, 2.5818, 2.7767, 2.9885, 3.2182, 3.4679, 3.7391, 4.0349,
01670
01671
01672
                4.3554, 4.7053, 5.0849, 5.4986, 5.9436, 6.4294, 6.9598,
               8.143, 8.8253, 9.5568, 10.371, 11.267, 12.233, 13.31, 14.357, 15.598, 16.93, 18.358, 19.849, 21.408, 23.04, 24.706, 26.409,
01674
01675
               28.153, 28.795, 30.549, 32.43, 34.49, 36.027, 38.955, 42.465, 46.565, 50.875, 55.378, 59.002, 63.882, 67.949, 73.693, 80.095, 86.403, 94.264, 102.65, 112.37, 123.3, 135.54, 149.14, 163.83, 179.17, 196.89, 217.91, 240.94, 264.13, 292.39, 324.83, 358.21,
01676
01677
01678
                397.16, 440.5, 488.6, 541.04, 595.3, 650.43, 652.03, 688.74,
01680
                719.47, 743.54, 757.68, 762.35, 756.43, 741.42, 595.43, 580.97, 580.83, 605.68, 667.88, 764.49, 759.93, 789.12, 798.17, 645.66,
01681
01682
                615.65, 455.05, 421.09, 306.45, 289.14, 235.7, 215.52, 274.57,
01683
                316.53, 357.73, 409.89, 465.06, 521.84, 579.02, 630.64, 794.46,
01684
                813., 813.56, 796.25, 761.57, 727.97, 812.14, 866.75, 932.5,
                1132.8, 1194.8, 1362.2, 1387.2, 1482.3, 1479.7, 1517.9, 1533.1
01687
                1534.2, 1523.3, 1522.5, 1515.5, 1505.2, 1486.5, 1454., 1412.,
01688
                1358.8, 1107.8, 1060.9, 1033.5, 1048.2, 1122.4, 1248.9, 1227.1,
                1255.4, 1058.9, 1020.7, 970.59, 715.24, 512.56, 468.47, 349.3, 338.26, 299.22, 301.26, 332.38, 382.08, 445.49, 515.87, 590.85,
01689
01690
```

```
662.3, 726.05, 955.59, 964.11, 945.17, 891.48, 807.11, 720.9,
              803.36, 834.46, 1073.9, 1107.1, 1123.6, 1296., 1393.7, 1303.1, 1284.3, 1161.8, 1078.8, 976.13, 868.72, 767.4, 674.72, 593.73,
01693
              523.12, 462.24, 409.75, 364.34, 325., 290.73, 260.76, 234.46,
01694
              211.28, 190.78, 172.61, 156.44, 142.01, 129.12, 117.57, 107.2, 97.877, 89.47, 81.882, 75.021, 68.807, 63.171, 58.052, 53.396, 49.155, 45.288, 41.759, 38.531, 35.576, 32.868, 30.384, 28.102, 26.003, 24.071, 22.293, 20.655, 19.147, 17.756, 16.476, 15.292,
01695
01696
01697
01698
              14.198, 13.183, 12.241, 11.367, 10.554, 9.7989, 9.0978, 8.4475, 7.845, 7.2868, 6.7704, 6.2927, 5.8508, 5.4421, 5.064, 4.714, 4.3902, 4.0902, 3.8121, 3.5543, 3.315, 3.093, 2.8869, 2.6953,
01699
01700
01701
01702
              2.5172, 2.3517, 2.1977, 2.0544, 1.9211, 1.7969, 1.6812, 1.5735,
01703
              1.4731, 1.3794, 1.2921, 1.2107, 1.1346, 1.0637, .99744, .93554,
              87771, .82368, .77313, .72587, .6816, .64014, .60134, .565, .53086, .49883, .46881, .44074, .4144, .38979, .36679, .34513,
01704
01705
              .32474, .30552, .28751, .27045, .25458, .23976, .22584, .21278, .20051, .18899, .17815, .16801, .15846, .14954, .14117, .13328,
01706
01707
01708
               .12584
01709
01710
01711
           double xw, dw, ew, cw296, cw260, cw230, dt230, dt260, dt296, ctw, ctmpth;
01712
01713
           int iw, jw;
01714
01715
            /* Get CO2 continuum absorption... */
01716
           xw = nu / 2 + 1;
01717
           if (xw >= 1 && xw < 2001) {
01718
            iw = (int) xw;
              jw = iw + 1;

dw = xw - iw;
01719
01720
01721
              ew = 1 - dw;
              cw296 = ew * co2296[iw - 1] + dw * co2296[jw - 1];
cw260 = ew * co2260[iw - 1] + dw * co2260[jw - 1];
01722
01723
              cw230 = ew * co2230[iw - 1] + dw * co2230[jw - 1];
01724
01725
              dt230 = t - 230;
              dt260 = t - 260;
01726
              dt296 = t - 296;
01727
              ctw = dt260 * 5.050505e-4 * dt296 * cw230 - dt230 * 9.259259e-4
  * dt296 * cw260 + dt230 * 4.208754e-4 * dt260 * cw296;
01729
01730
               ctmpth = u / GSL_CONST_NUM_AVOGADRO / 1000 * p / P0 * ctw;
01731
              ctmpth = 0;
01732
01733
            return ctmpth;
01734 }
```

5.15.2.7 double ctmh2o ( double nu, double p, double t, double q, double u )

Compute water vapor continuum (optical depth).

Definition at line 1738 of file jurassic.c.

```
01743
                      {
01744
         static double h2o296[2001] = { .17, .1695, .172, .168, .1687, .1624, .1606,
           .1508, .1447, .1344, .1214, .1133, .1009, .09217, .08297, .06989,
01746
            .06513, .05469, .05056, .04417, .03779, .03484, .02994,
01747
01748
            .02325, .02063, .01818, .01592, .01405, .01251, .0108,
01749
            .008424, .007519, .006555, .00588, .005136, .004511, .003989,
            .003509, .003114, .00274, .002446, .002144, .001895, .001676, .001486, .001312, .001164, .001031, 9.129e-4, 8.106e-4, 7.213e-4, 6.4e-4, 5.687e-4, 5.063e-4, 4.511e-4, 4.029e-4, 3.596e-4,
01750
01751
01753
            3.22e-4, 2.889e-4, 2.597e-4, 2.337e-4, 2.108e-4, 1.907e-4
            1.728e-4, 1.57e-4, 1.43e-4, 1.305e-4, 1.195e-4, 1.097e-4,
01754
01755
            1.009e-4, 9.307e-5, 8.604e-5, 7.971e-5, 7.407e-5, 6.896e-5,
            6.433e-5, 6.013e-5, 5.631e-5, 5.283e-5, 4.963e-5, 4.669e-5,
01756
01757
            4.398e-5, 4.148e-5, 3.917e-5, 3.702e-5, 3.502e-5, 3.316e-5,
            3.142e-5, 2.978e-5, 2.825e-5, 2.681e-5, 2.546e-5, 2.419e-5,
            2.299e-5, 2.186e-5, 2.079e-5, 1.979e-5, 1.884e-5, 1.795e-5,
01759
01760
            1.711e-5, 1.633e-5, 1.559e-5, 1.49e-5, 1.426e-5, 1.367e-5
01761
            1.312e-5, 1.263e-5, 1.218e-5, 1.178e-5, 1.143e-5, 1.112e-5,
            1.088e-5, 1.07e-5, 1.05re-5, 1.05e-5, 1.051e-5, 1.059e-5, 1.076e-5, 1.1e-5, 1.133e-5, 1.18e-5, 1.237e-5, 1.308e-5,
01762
01763
            1.393e-5, 1.483e-5, 1.614e-5, 1.758e-5, 1.93e-5, 2.123e-5,
01765
            2.346e-5, 2.647e-5, 2.93e-5, 3.279e-5, 3.745e-5, 4.152e-5,
01766
            4.813e-5, 5.477e-5, 6.203e-5, 7.331e-5, 8.056e-5, 9.882e-5,
01767
            1.05e-4, 1.21e-4, 1.341e-4, 1.572e-4, 1.698e-4, 1.968e-4,
            2.175e-4, 2.431e-4, 2.735e-4, 2.867e-4, 3.19e-4, 3.371e-4, 3.554e-4, 3.726e-4, 3.837e-4, 3.878e-4, 3.864e-4, 3.858e-4, 3.841e-4, 3.852e-4, 3.815e-4, 3.762e-4, 3.618e-4, 3.579e-4,
01768
01769
01770
01771
            3.45e-4, 3.202e-4, 3.018e-4, 2.785e-4, 2.602e-4, 2.416e-4,
```

```
2.097e-4, 1.939e-4, 1.689e-4, 1.498e-4, 1.308e-4, 1.17e-4,
            1.011e-4, 9.237e-5, 7.909e-5, 7.006e-5, 6.112e-5, 5.401e-5, 4.914e-5, 4.266e-5, 3.963e-5, 3.316e-5, 3.037e-5, 2.598e-5,
01773
01774
            2.294e-5, 2.066e-5, 1.813e-5, 1.583e-5, 1.423e-5, 1.247e-5, 1.116e-5, 9.76e-6, 8.596e-6, 7.72e-6, 6.825e-6, 6.108e-6,
01775
01776
            1.11de-3, 9.7de-6, 0.13de-6, 1.22e-6, 0.22e-6, 0.23e-6, 0.13de-6, 2.972e-6, 3.731e-6, 3.346e-6, 2.972e-6, 2.628e-6, 2.356e-6, 2.102e-6, 1.878e-6, 1.678e-6, 1.507e-6,
01777
01778
            1.348e-6, 1.21e-6, 1.089e-6, 9.806e-7, 8.857e-7, 8.004e-7,
01779
            7.261e-7, 6.599e-7, 6.005e-7, 5.479e-7, 5.011e-7, 4.595e-7
01780
            4.219e-7, 3.885e-7, 3.583e-7, 3.314e-7, 3.071e-7, 2.852e-7,
01781
            2.654e-7, 2.474e-7, 2.311e-7, 2.162e-7, 2.026e-7, 1.902e-7, 1.788e-7, 1.683e-7, 1.587e-7, 1.497e-7, 1.415e-7, 1.338e-7,
01782
01783
01784
             1.266e-7, 1.2e-7, 1.138e-7, 1.08e-7, 1.027e-7, 9.764e-8,
            9.296e-8, 8.862e-8, 8.458e-8, 8.087e-8, 7.744e-8, 7.429e-8,
01785
01786
            7.145e-8, 6.893e-8, 6.664e-8, 6.468e-8, 6.322e-8, 6.162e-8,
01787
            6.07e-8, 5.992e-8, 5.913e-8, 5.841e-8, 5.796e-8, 5.757e-8,
            5.746e-8, 5.731e-8, 5.679e-8, 5.577e-8, 5.671e-8, 5.656e-8, 5.594e-8, 5.593e-8, 5.602e-8, 5.62e-8, 5.693e-8, 5.725e-8, 5.858e-8, 6.037e-8, 6.249e-8, 6.535e-8, 6.899e-8, 7.356e-8,
01788
01789
01791
             7.918e-8, 8.618e-8, 9.385e-8, 1.039e-7, 1.158e-7, 1.29e-7,
            1.437e-7, 1.65e-7, 1.871e-7, 2.121e-7, 2.427e-7, 2.773e-7, 3.247e-7, 3.677e-7, 4.037e-7, 4.776e-7, 5.101e-7, 6.214e-7,
01792
01793
            6.936e-7, 7.581e-7, 8.486e-7, 9.355e-7, 9.942e-7, 1.063e-6,
01794
01795
            1.123e-6, 1.191e-6, 1.215e-6, 1.247e-6, 1.26e-6, 1.271e-6,
01796
            1.284e-6, 1.317e-6, 1.323e-6, 1.349e-6, 1.353e-6, 1.362e-6,
01797
            1.344e-6, 1.329e-6, 1.336e-6, 1.327e-6, 1.325e-6, 1.359e-6,
01798
            1.374e-6, 1.415e-6, 1.462e-6, 1.526e-6, 1.619e-6, 1.735e-6,
01799
            1.863e-6, 2.034e-6, 2.265e-6, 2.482e-6, 2.756e-6, 3.103e-6,
01800
            3.466e-6, 3.832e-6, 4.378e-6, 4.913e-6, 5.651e-6, 6.311e-6,
01801
            7.169e-6, 8.057e-6, 9.253e-6, 1.047e-5, 1.212e-5, 1.36e-5,
            1.569e-5, 1.776e-5, 2.02e-5, 2.281e-5, 2.683e-5, 2.994e-5,
01802
01803
            3.488e-5, 3.896e-5, 4.499e-5, 5.175e-5, 6.035e-5, 6.34e-5,
            7.281e-5, 7.923e-5, 8.348e-5, 9.631e-5, 1.044e-4, 1.102e-4,
01804
01805
            1.176e-4, 1.244e-4, 1.283e-4, 1.326e-4, 1.4e-4, 1.395e-4,
            1.387e-4, 1.363e-4, 1.314e-4, 1.241e-4, 1.228e-4, 1.148e-4, 1.086e-4, 1.018e-4, 8.89e-5, 8.316e-5, 7.292e-5, 6.452e-5, 5.625e-5, 5.045e-5, 4.38e-5, 3.762e-5, 3.29e-5, 2.836e-5,
01806
01807
01808
            2.485e-5, 2.168e-5, 1.895e-5, 1.659e-5, 1.453e-5, 1.282e-5,
01810
            1.132e-5, 1.001e-5, 8.836e-6, 7.804e-6, 6.922e-6, 6.116e-6,
            5.429e-6, 4.824e-6, 4.278e-6, 3.788e-6, 3.371e-6, 2.985e-6,
01811
01812
            2.649e-6, 2.357e-6, 2.09e-6, 1.858e-6, 1.647e-6, 1.462e-6,
            1.299e-6, 1.155e-6, 1.028e-6, 9.142e-7, 8.132e-7, 7.246e-7,
01813
            3.318e-7, 2.985e-7, 2.69e-7, 2.428e-7, 2.197e-7, 1.992e-7,
01814
01815
            1.81e-7, 1.649e-7, 1.506e-7, 1.378e-7, 1.265e-7, 1.163e-7,
01817
            1.073e-7, 9.918e-8, 9.191e-8, 8.538e-8, 7.949e-8, 7.419e-8,
01818
            6.94e-8, 6.508e-8, 6.114e-8, 5.761e-8, 5.437e-8, 5.146e-8,
01819
            4.89e-8, 4.636e-8, 4.406e-8, 4.201e-8, 4.015e-8, 3.84e-8,
            3.661e-8, 3.51e-8, 3.377e-8, 3.242e-8, 3.13e-8, 3.015e-8, 2.918e-8, 2.83e-8, 2.758e-8, 2.707e-8, 2.656e-8, 2.619e-8,
01820
01821
            2.609e-8, 2.615e-8, 2.63e-8, 2.675e-8, 2.745e-8, 2.842e-8,
            2.966e-8, 3.125e-8, 3.318e-8, 3.565e-8, 3.85e-8, 4.191e-8,
01823
01824
            4.59e-8, 5.059e-8, 5.607e-8, 6.239e-8, 6.958e-8, 7.796e-8,
            8.773e-8, 9.88e-8, 1.114e-7, 1.258e-7, 1.422e-7, 1.61e-7, 1.822e-7, 2.06e-7, 2.337e-7, 2.645e-7, 2.996e-7, 3.393e-7,
01825
01826
            3.843e-7, 4.363e-7, 4.935e-7, 5.607e-7, 6.363e-7, 7.242e-7,
01827
            8.23e-7, 9.411e-7, 1.071e-6, 1.232e-6, 1.402e-6, 1.6e-6, 1.82e-6,
            2.128e-6, 2.386e-6, 2.781e-6, 3.242e-6, 3.653e-6, 4.323e-
01829
            4.747e-6, 5.321e-6, 5.919e-6, 6.681e-6, 7.101e-6, 7.983e-6,
01830
01831
            8.342e-6, 8.741e-6, 9.431e-6, 9.952e-6, 1.026e-5, 1.055e-5,
01832
            1.095e-5, 1.095e-5, 1.087e-5, 1.056e-5, 1.026e-5, 9.715e-6,
01833
            9.252e-6, 8.452e-6, 7.958e-6, 7.268e-6, 6.295e-6, 6.003e-6, 5e-6,
            4.591e-6, 3.983e-6, 3.479e-6, 3.058e-6, 2.667e-6, 2.293e-6,
            1.995e-6, 1.747e-6, 1.517e-6, 1.335e-6, 1.165e-6, 1.028e-6,
01836
            9.007e-7, 7.956e-7, 7.015e-7, 6.192e-7, 5.491e-7, 4.859e-7,
            4.297e-7, 3.799e-7, 3.38e-7, 3.002e-7, 2.659e-7, 2.366e-7, 2.103e-7, 1.861e-7, 1.655e-7, 1.469e-7, 1.309e-7, 1.162e-7, 1.032e-7, 9.198e-8, 8.181e-8, 7.294e-8, 6.516e-8, 5.787e-8,
01837
01838
01839
01840
            5.163e-8, 4.612e-8, 4.119e-8, 3.695e-8, 3.308e-8, 2.976e-8,
            2.67e-8, 2.407e-8, 2.171e-8, 1.965e-8, 1.78e-8, 1.617e-8,
            1.47e-8, 1.341e-8, 1.227e-8, 1.125e-8, 1.033e-8, 9.524e-9,
01842
01843
            8.797e-9, 8.162e-9, 7.565e-9, 7.04e-9, 6.56e-9, 6.129e-9,
01844
            5.733e-9, 5.376e-9, 5.043e-9, 4.75e-9, 4.466e-9, 4.211e-9,
            3.977e-9, 3.759e-9, 3.558e-9, 3.373e-9, 3.201e-9, 3.043e-9, 2.895e-9, 2.76e-9, 2.635e-9, 2.518e-9, 2.411e-9, 2.314e-9, 2.23e-9, 2.151e-9, 2.087e-9, 2.035e-9, 1.988e-9, 1.946e-9,
01845
01846
            1.927e-9, 1.916e-9, 1.916e-9, 1.933e-9, 1.966e-9, 2.018e-9,
01848
01849
            2.09e-9, 2.182e-9, 2.299e-9, 2.442e-9, 2.623e-9, 2.832e-9,
            3.079e-9, 3.368e-9, 3.714e-9, 4.104e-9, 4.567e-9, 5.091e-9, 5.701e-9, 6.398e-9, 7.194e-9, 8.127e-9, 9.141e-9, 1.035e-8,
01850
01851
            1.177e-8, 1.338e-8, 1.508e-8, 1.711e-8, 1.955e-8, 2.216e-8,
01852
            2.534e-8, 2.871e-8, 3.291e-8, 3.711e-8, 4.285e-8, 4.868e-8,
            5.509e-8, 6.276e-8, 7.262e-8, 8.252e-8, 9.4e-8, 1.064e-7, 1.247e-7, 1.411e-7, 1.626e-7, 1.827e-7, 2.044e-7, 2.284e-7,
01854
01855
01856
            2.452e-7, 2.854e-7, 3.026e-7, 3.278e-7, 3.474e-7, 3.693e-7,
            3.93e-7, 4.104e-7, 4.22e-7, 4.439e-7, 4.545e-7, 4.778e-7, 4.812e-7, 5.018e-7, 4.899e-7, 5.075e-7, 5.073e-7, 5.171e-7,
01857
01858
```

```
5.131e-7, 5.25e-7, 5.617e-7, 5.846e-7, 6.239e-7, 6.696e-7,
                  7.398e-7, 8.073e-7, 9.15e-7, 1.009e-6, 1.116e-6, 1.264e-6, 1.439e-6, 1.644e-6, 1.856e-6, 2.147e-6, 2.317e-6, 2.713e-6,
01860
01861
                  2.882e-6, 2.99e-6, 3.489e-6, 3.581e-6, 4.033e-6, 4.26e-6, 4.543e-6, 4.84e-6, 4.826e-6, 5.013e-6, 5.252e-6, 5.277e-6, 5.306e-6, 5.236e-6, 5.123e-6, 5.171e-6, 4.843e-6, 4.615e-6,
01862
01863
01864
                  4.385e-6, 3.97e-6, 3.693e-6, 3.231e-6, 2.915e-6, 2.495e-6, 2.144e-6, 1.91e-6, 1.639e-6, 1.417e-6, 1.226e-6, 1.065e-6,
01866
                  2.134e 0, 1.31e 0, 1.635e 0, 1.141e 0, 1.226e 1, 1.605e 0, 1.605e 
01867
01868
01869
01870
                  5.216e-8, 4.632e-8, 4.127e-8, 3.678e-8, 3.279e-8, 2.923e-8,
01871
                  2.612e-8, 2.339e-8, 2.094e-8, 1.877e-8, 1.686e-8, 1.516e-8,
01872
01873
                  1.366e-8, 1.234e-8, 1.114e-8, 1.012e-8, 9.182e-9, 8.362e-9,
                  7.634e-9, 6.981e-9, 6.406e-9, 5.888e-9, 5.428e-9, 5.021e-9, 4.65e-9, 4.326e-9, 4.033e-9, 3.77e-9, 3.536e-9, 3.327e-9, 3.141e-9, 2.974e-9, 2.825e-9, 2.697e-9, 2.584e-9, 2.488e-9,
01874
01875
01876
                  2.406e-9, 2.34e-9, 2.292e-9, 2.259e-9, 2.244e-9, 2.243e-9, 2.272e-9, 2.31e-9, 2.378e-9, 2.454e-9, 2.618e-9, 2.672e-9,
01878
                  2.831e-9, 3.05e-9, 3.225e-9, 3.425e-9, 3.67re-9, 3.968e-9, 4.221e-9, 4.639e-9, 4.96e-9, 5.359e-9, 5.649e-9, 6.23e-9,
01879
01880
                  6.716e-9, 7.218e-9, 7.746e-9, 7.988e-9, 8.627e-9, 8.999e-9,
01881
                  9.442e-9, 9.82e-9, 1.015e-8, 1.06e-8, 1.079e-8, 1.109e-8, 1.137e-8, 1.186e-8, 1.18e-8, 1.187e-8, 1.194e-8, 1.192e-8,
01882
01883
                  1.224e-8, 1.245e-8, 1.246e-8, 1.318e-8, 1.377e-8, 1.471e-8,
01885
                  1.582e-8, 1.713e-8, 1.853e-8, 2.063e-8, 2.27e-8, 2.567e-8,
01886
                  2.891e-8, 3.264e-8, 3.744e-8, 4.286e-8, 4.915e-8, 5.623e-8,
                  2.031e-0, 3.201e-0, 3.741e-0, 4.200e-0, 4.313e-0, 3.623e-7, 3.633e-8, 7.293e-8, 8.309e-8, 9.319e-8, 1.091e-7, 1.243e-7, 1.348e-7, 1.449e-7, 1.62e-7, 1.846e-7, 1.937e-7, 2.04e-7, 2.179e-7, 2.298e-7, 2.433e-7, 2.439e-7, 2.464e-7, 2.611e-7, 2.617e-7, 2.582e-7, 2.453e-7, 2.401e-7, 2.349e-7, 2.203e-7,
01887
01888
01889
01890
                  2.066e-7, 1.939e-7, 1.78e-7, 1.558e-7, 1.391e-7, 1.203e-7,
01891
01892
                  1.048e-7, 9.464e-8, 8.306e-8, 7.239e-8, 6.317e-8, 5.52e-8
                  4.847e-8, 4.282e-8, 3.796e-8, 3.377e-8, 2.996e-8, 2.678e-8, 2.4e-8, 2.134e-8, 1.904e-8, 1.705e-8, 1.523e-8, 1.35e-8, 1.204e-8, 1.07e-8, 9.408e-9, 8.476e-9, 7.47e-9, 6.679e-9,
01893
01894
01895
                   5.929e-9, 5.267e-9, 4.711e-9, 4.172e-9, 3.761e-9, 3.288e-9,
01897
                  2.929e-9, 2.609e-9, 2.315e-9, 2.042e-9, 1.844e-9, 1.64e-9,
01898
                   1.47e-9, 1.31e-9, 1.176e-9, 1.049e-9, 9.377e-10, 8.462e-10,
01899
                  7.616e-10, 6.854e-10, 6.191e-10, 5.596e-10, 5.078e-10, 4.611e-10,
                  4.197e-10, 3.83e-10, 3.505e-10, 3.215e-10, 2.956e-10, 2.726e-10, 2.521e-10, 2.338e-10, 2.173e-10, 2.026e-10, 1.895e-10, 1.777e-10, 1.672e-10, 1.579e-10, 1.496e-10, 1.423e-10, 1.358e-10, 1.302e-10,
01900
01901
                  1.254e-10, 1.216e-10, 1.187e-10, 1.163e-10, 1.147e-10, 1.145e-10,
01903
01904
                  1.15e-10, 1.17e-10, 1.192e-10, 1.25e-10, 1.298e-10, 1.345e-10,
01905
                  1.405e-10, 1.538e-10, 1.648e-10, 1.721e-10, 1.872e-10, 1.968e-10,
                  2.089e-10, 2.172e-10, 2.317e-10, 2.389e-10, 2.503e-10, 2.585e-10, 2.686e-10, 2.8e-10, 2.895e-10, 3.019e-10, 3.037e-10, 3.076e-10,
01906
01907
                  3.146e-10, 3.198e-10, 3.332e-10, 3.397e-10, 3.54e-10, 3.667e-10,
01908
                  3.895e-10, 4.071e-10, 4.565e-10, 4.983e-10, 5.439e-10, 5.968e-10,
                  6.676e-10, 7.456e-10, 8.405e-10, 9.478e-10, 1.064e-9, 1.218e-9,
01910
01911
                  1.386e-9, 1.581e-9, 1.787e-9, 2.032e-9, 2.347e-9, 2.677e-9,
01912
                  3.008e-9, 3.544e-9, 4.056e-9, 4.687e-9, 5.331e-9, 6.227e-9,
                  6.854e-9, 8.139e-9, 8.945e-9, 9.865e-9, 1.125e-8, 1.178e-8,
01913
                  1.364e-8, 1.436e-8, 1.54e-8, 1.672e-8, 1.793e-8, 1.906e-8,
01914
                  2.036e-8, 2.144e-8, 2.292e-8, 2.371e-8, 2.493e-8, 2.606e-8,
01916
                  2.706e-8, 2.866e-8, 3.036e-8, 3.136e-8, 3.405e-8, 3.665e-8,
01917
                  3.837e-8, 4.229e-8, 4.748e-8, 5.32e-8, 5.763e-8, 6.677e-8,
                  7.216e-8, 7.716e-8, 8.958e-8, 9.419e-8, 1.036e-7, 1.108e-7,
01918
                  1.189e-7, 1.246e-7, 1.348e-7, 1.31e-7, 1.361e-7, 1.364e-7, 1.363e-7, 1.343e-7, 1.254e-7, 1.235e-7, 1.158e-7,
01919
01920
                   1.107e-7, 9.961e-8, 9.011e-8, 7.91e-8, 6.916e-8, 6.338e-8,
                  5.564e-8, 4.827e-8, 4.198e-8, 3.695e-8, 3.276e-8, 2.929e-8,
01922
01923
                  2.633e-8, 2.391e-8, 2.192e-8, 2.021e-8, 1.89e-8, 1.772e-8,
01924
                  1.667e-8, 1.603e-8, 1.547e-8, 1.537e-8, 1.492e-8, 1.515e-8,
01925
                  1.479e-8, 1.45e-8, 1.513e-8, 1.495e-8, 1.529e-8, 1.565e-8, 1.564e-8, 1.553e-8, 1.569e-8, 1.584e-8, 1.57e-8, 1.538e-8,
01926
                  1.513e-8, 1.472e-8, 1.425e-8, 1.349e-8, 1.328e-8, 1.249e-8,
01927
                   1.17e-8, 1.077e-8, 9.514e-9, 8.614e-9, 7.46e-9, 6.621e-9,
                  5.775e-9, 5.006e-9, 4.308e-9, 3.747e-9, 3.24e-9, 2.84e-9, 2.481e-9, 2.184e-9, 1.923e-9, 1.71e-9, 1.504e-9, 1.334e-9,
01929
01930
                  1.187e-9, 1.053e-9, 9.367e-10, 8.306e-10, 7.419e-10, 6.63e-10, 5.918e-10, 5.277e-10, 4.717e-10, 4.222e-10, 3.783e-10, 3.39e-10, 3.036e-10, 2.729e-10, 2.455e-10, 2.211e-10, 1.995e-10, 1.804e-10,
01931
01932
01933
                  1.635e-10, 1.485e-10, 1.355e-10, 1.24e-10, 1.139e-10, 1.051e-10,
01934
01935
                  9.757e-11, 9.114e-11, 8.577e-11, 8.139e-11, 7.792e-11, 7.52e-11,
                  7.39e-11, 7.311e-11, 7.277e-11, 7.482e-11, 7.698e-11, 8.162e-11,
01936
                  8.517e-11, 8.968e-11, 9.905e-11, 1.075e-10, 1.187e-10, 1.291e-10,
01937
                  1.426e-10, 1.573e-10, 1.734e-10, 1.905e-10, 2.097e-10, 2.28e-10,
01938
                  2.473e-10, 2.718e-10, 2.922e-10, 3.128e-10, 3.361e-10, 3.641e-10, 3.91e-10, 4.196e-10, 4.501e-10, 4.932e-10, 5.258e-10, 5.755e-10,
01939
01941
                   6.253e-10, 6.664e-10, 7.344e-10, 7.985e-10, 8.877e-10, 1.005e-9,
01942
                  1.118e-9, 1.251e-9, 1.428e-9, 1.61e-9, 1.888e-9, 2.077e-9,
01943
                  2.331e-9, 2.751e-9, 3.061e-9, 3.522e-9, 3.805e-9, 4.181e-9,
                  4.575e-9, 5.167e-9, 5.634e-9, 6.007e-9, 6.501e-9, 6.829e-9, 7.211e-9, 7.262e-9, 7.696e-9, 7.832e-9, 7.799e-9, 7.651e-9,
01944
01945
```

```
7.304e-9, 7.15e-9, 6.977e-9, 6.603e-9, 6.209e-9, 5.69e-9,
           5.432e-9, 4.764e-9, 4.189e-9, 3.64e-9, 3.203e-9, 2.848e-9, 2.51e-9, 2.194e-9, 1.946e-9, 1.75e-9, 1.567e-9, 1.426e-9,
01947
01948
            1.302e-9, 1.197e-9, 1.109e-9, 1.035e-9, 9.719e-10, 9.207e-10,
01949
            8.957e-10, 8.578e-10, 8.262e-10, 8.117e-10, 7.987e-10, 7.875e-10,
01950
            7.741e-10, 7.762e-10, 7.537e-10, 7.424e-10, 7.474e-10, 7.294e-10,
01951
            7.216e-10, 7.233e-10, 7.075e-10, 6.892e-10, 6.618e-10, 6.314e-10,
            6.208e-10, 5.689e-10, 5.55e-10, 4.984e-10, 4.6e-10, 4.078e-10,
01953
01954
            3.879e-10, 3.459e-10, 2.982e-10, 2.626e-10, 2.329e-10, 1.988e-10,
01955
            1.735e-10, 1.487e-10, 1.297e-10, 1.133e-10, 9.943e-11, 8.736e-11,
            7.726e-11, 6.836e-11, 6.053e-11, 5.384e-11, 4.789e-11, 4.267e-11,
01956
           3.804e-11, 3.398e-11, 3.034e-11, 2.71e-11, 2.425e-11, 2.173e-11, 1.95e-11, 1.752e-11, 1.574e-11, 1.418e-11, 1.278e-11, 1.154e-11,
01957
01958
            1.044e-11, 9.463e-12, 8.602e-12, 7.841e-12, 7.171e-12, 6.584e-12,
01959
01960
            6.073e-12, 5.631e-12, 5.254e-12, 4.937e-12, 4.679e-12, 4.476e-12,
01961
            4.328e-12, 4.233e-12, 4.194e-12, 4.211e-12, 4.286e-12, 4.424e-12,
01962
            4.628e-12, 4.906e-12, 5.262e-12, 5.708e-12, 6.254e-12, 6.914e-12,
            7.714e-12, 8.677e-12, 9.747e-12, 1.101e-11, 1.256e-11, 1.409e-11,
01963
01964
            1.597e-11, 1.807e-11, 2.034e-11, 2.316e-11, 2.622e-11, 2.962e-11,
            3.369e-11, 3.819e-11, 4.329e-11, 4.932e-11, 5.589e-11, 6.364e-11,
            7.284e-11, 8.236e-11, 9.447e-11, 1.078e-10, 1.229e-10, 1.417e-10,
01966
01967
            1.614e-10, 1.843e-10, 2.107e-10, 2.406e-10, 2.728e-10, 3.195e-10,
01968
            3.595e-10, 4.153e-10, 4.736e-10, 5.41e-10, 6.088e-10, 6.769e-10,
            7.691e-10, 8.545e-10, 9.621e-10, 1.047e-9, 1.161e-9, 1.296e-9,
01969
            1.424e-9, 1.576e-9, 1.739e-9, 1.893e-9, 2.08e-9, 2.336e-9,
01970
            2.604e-9, 2.76e-9, 3.001e-9, 3.365e-9, 3.55e-9, 3.895e-9,
01971
            4.183e-9, 4.614e-9, 4.846e-9, 5.068e-9, 5.427e-9, 5.541e-9,
01972
01973
            5.864e-9, 5.997e-9, 5.997e-9, 6.061e-9, 5.944e-9, 5.855e-9,
01974
            5.661e-9, 5.523e-9, 5.374e-9, 4.94e-9, 4.688e-9, 4.17e-9,
            3.913e-9, 3.423e-9, 2.997e-9, 2.598e-9, 2.253e-9, 1.946e-9, 1.71e-9, 1.507e-9, 1.336e-9, 1.19e-9, 1.068e-9, 9.623e-10,
01975
01976
            8.772e-10, 8.007e-10, 7.42e-10, 6.884e-10, 6.483e-10, 6.162e-10,
            5.922e-10, 5.688e-10, 5.654e-10, 5.637e-10, 5.701e-10, 5.781e-10,
01978
01979
            5.874e-10, 6.268e-10, 6.357e-10, 6.525e-10, 7.137e-10, 7.441e-10,
           8.024e-10, 8.485e-10, 9.143e-10, 9.536e-10, 9.717e-10, 1.018e-9, 1.042e-9, 1.054e-9, 1.092e-9, 1.079e-9, 1.064e-9, 1.043e-9, 1.02e-9, 9.687e-10, 9.273e-10, 9.208e-10, 9.068e-10, 7.687e-10, 7.385e-10, 6.595e-10, 5.87e-10, 5.144e-10, 4.417e-10, 3.804e-10,
01980
01981
01982
01984
            3.301e-10, 2.866e-10, 2.509e-10, 2.202e-10, 1.947e-10, 1.719e-10,
            1.525e-10, 1.361e-10, 1.21e-10, 1.084e-10, 9.8e-11, 8.801e-11,
01985
01986
            7.954e-11,\ 7.124e-11,\ 6.335e-11,\ 5.76e-11,\ 5.132e-11,\ 4.601e-11,
           4.096e-11, 3.657e-11, 3.25e-11, 2.909e-11, 2.587e-11, 2.297e-11, 2.05e-11, 1.828e-11, 1.632e-11, 1.462e-11, 1.314e-11, 1.185e-11, 1.073e-11, 9.76e-12, 8.922e-12, 8.206e-12, 7.602e-12, 7.1e-12,
01987
01988
            6.694e-12, 6.378e-12, 6.149e-12, 6.004e-12, 5.941e-12, 5.962e-12,
01990
01991
            6.069e-12, 6.265e-12, 6.551e-12, 6.935e-12, 7.457e-12, 8.074e-12,
01992
            8.811e-12, 9.852e-12, 1.086e-11, 1.207e-11, 1.361e-11, 1.553e-11,
01993
           1.737 e-11, \ 1.93 e-11, \ 2.175 e-11, \ 2.41 e-11, \ 2.706 e-11, \ 3.023 e-11,
            3.313e-11, 3.657e-11, 4.118e-11, 4.569e-11, 5.025e-11, 5.66e-11, 6.231e-11, 6.881e-11, 7.996e-11, 8.526e-11, 9.694e-11, 1.106e-10,
01994
01995
            1.222e-10, 1.355e-10, 1.525e-10, 1.775e-10, 1.924e-10, 2.181e-10,
            2.379e-10, 2.662e-10, 2.907e-10, 3.154e-10, 3.366e-10, 3.579e-10,
01997
01998
            3.858e-10, 4.046e-10, 4.196e-10, 4.166e-10, 4.457e-10, 4.466e-10,
           4.404e-10, 4.337e-10, 4.15e-10, 4.083e-10, 3.91e-10, 3.723e-10, 3.514e-10, 3.303e-10, 2.847e-10, 2.546e-10, 2.23e-10, 1.994e-10,
01999
02000
            1.733e-10, 1.488e-10, 1.297e-10, 1.144e-10, 1.004e-10, 8.741e-11,
02001
            7.928e-11, 7.034e-11, 6.323e-11, 5.754e-11, 5.25e-11, 4.85e-11,
            4.502e-11, 4.286e-11, 4.028e-11, 3.899e-11, 3.824e-11, 3.761e-11,
02003
            3.804e-11, 3.839e-11, 3.845e-11, 4.244e-11, 4.382e-11, 4.582e-11,
02004
02005
            4.847e-11, 5.209e-11, 5.384e-11, 5.887e-11, 6.371e-11, 6.737e-11,
02006
            7.168e-11,\ 7.415e-11,\ 7.827e-11,\ 8.037e-11,\ 8.12e-11,\ 8.071e-11,
           8.008e-11, 7.851e-11, 7.544e-11, 7.377e-11, 7.173e-11, 6.801e-11, 6.267e-11, 5.727e-11, 5.288e-11, 4.853e-11, 4.082e-11, 3.645e-11,
02007
02008
            3.136e-11, 2.672e-11, 2.304e-11, 1.986e-11, 1.725e-11, 1.503e-11,
02009
02010
            1.315e-11, 1.153e-11, 1.014e-11, 8.942e-12, 7.901e-12, 6.993e-12,
02011
            6.199e-12, 5.502e-12, 4.89e-12, 4.351e-12, 3.878e-12, 3.461e-12,
02012
            3.094e-12, 2.771e-12, 2.488e-12, 2.241e-12, 2.025e-12, 1.838e-12,
            1.677e-12, 1.541e-12, 1.427e-12, 1.335e-12, 1.262e-12, 1.209e-12,
02013
           1.176e-12, 1.161e-12, 1.165e-12, 1.189e-12, 1.234e-12, 1.3e-12,
02014
            1.389e-12, 1.503e-12, 1.644e-12, 1.814e-12, 2.017e-12, 2.255e-12,
            2.534e-12, 2.858e-12, 3.231e-12, 3.661e-12, 4.153e-12, 4.717e-12, 5.36e-12, 6.094e-12, 6.93e-12, 7.882e-12, 8.966e-12, 1.02e-11,
02016
02017
           1.162e-11, 1.324e-11, 1.51e-11, 1.72e-11, 1.965e-11, 2.237e-11, 2.56e-11, 2.927e-11, 3.371e-11, 3.842e-11, 4.429e-11, 5.139e-11,
02018
02019
02020
            5.798e-11, 6.697e-11, 7.626e-11, 8.647e-11, 1.022e-10, 1.136e-10,
            1.3e-10, 1.481e-10, 1.672e-10, 1.871e-10, 2.126e-10, 2.357e-10,
02021
            2.583e-10, 2.997e-10, 3.289e-10, 3.702e-10, 4.012e-10, 4.319e-10,
02022
02023
            4.527e-10, 5.001e-10, 5.448e-10, 5.611e-10, 5.76e-10, 5.965e-10,
02024
            6.079e-10, 6.207e-10, 6.276e-10, 6.222e-10, 6.137e-10, 6e-10,
           5.814e-10, 5.393e-10, 5.35e-10, 4.947e-10, 4.629e-10, 4.117e-10, 3.712e-10, 3.372e-10, 2.923e-10, 2.55e-10, 2.232e-10, 1.929e-10, 1.679e-10, 1.46e-10, 1.289e-10, 1.13e-10, 9.953e-11, 8.763e-11,
02025
02026
            7.76e-11, 6.9e-11, 6.16e-11, 5.525e-11, 4.958e-11, 4.489e-11,
02028
02029
            4.072e-11, 3.728e-11, 3.438e-11, 3.205e-11, 3.006e-11, 2.848e-11,
02030
            2.766e-11,\ 2.688e-11,\ 2.664e-11,\ 2.67e-11,\ 2.696e-11,\ 2.786e-11,
            2.861e-11, 3.009e-11, 3.178e-11, 3.389e-11, 3.587e-11, 3.819e-11, 4.054e-11, 4.417e-11, 4.703e-11, 5.137e-11, 5.46e-11, 6.055e-11,
02031
02032
```

```
6.333e-11, 6.773e-11, 7.219e-11, 7.717e-11, 8.131e-11, 8.491e-11,
             8.574e-11, 9.01e-11, 9.017e-11, 8.999e-11, 8.959e-11, 8.838e-11,
02034
02035
             8.579e-11, 8.162e-11, 8.098e-11, 7.472e-11, 7.108e-11, 6.559e-11,
02036
             5.994 e-11, \ 5.172 e-11, \ 4.424 e-11, \ 3.951 e-11, \ 3.34 e-11, \ 2.902 e-11,
             2.54le-11, 2.215e-11, 1.945e-11, 1.716e-11, 1.503e-11, 1.339e-11, 1.185e-11, 1.05e-11, 9.336e-12, 8.307e-12, 7.312e-12, 6.55e-12,
02037
02038
             5.836e-12, 5.178e-12, 4.6e-12, 4.086e-12, 3.639e-12, 3.247e-12,
             2.904e-12, 2.604e-12, 2.341e-12, 2.112e-12, 1.914e-12, 1.744e-12,
02040
02041
             1.598e-12, 1.476e-12, 1.374e-12, 1.293e-12, 1.23e-12, 1.185e-12,
02042
             1.158e-12, 1.147e-12, 1.154e-12, 1.177e-12, 1.219e-12, 1.28e-12,
02043
             1.36e-12, 1.463e-12, 1.591e-12, 1.75e-12, 1.94e-12, 2.156e-12,
             2.43e-12, 2.748e-12, 3.052e-12, 3.533e-12, 3.967e-12, 4.471e-12, 5.041e-12, 5.86e-12, 6.664e-12, 7.522e-12, 8.342e-12, 9.412e-12,
02044
02045
             1.072e-11, 1.213e-11, 1.343e-11, 1.496e-11, 1.664e-11, 1.822e-11,
02046
02047
             2.029e-11, 2.233e-11, 2.457e-11, 2.709e-11, 2.928e-11, 3.115e-11,
02048
             3.356e-11, 3.592e-11, 3.818e-11, 3.936e-11, 4.061e-11, 4.149e-11,
             4.299e-11, 4.223e-11, 4.251e-11, 4.287e-11, 4.177e-11, 4.094e-11,
02049
             3.942e-11, 3.772e-11, 3.614e-11, 3.394e-11, 3.222e-11, 2.791e-11, 2.665e-11, 2.309e-11, 2.032e-11, 1.74e-11, 1.535e-11, 1.323e-11,
02050
             1.151e-11, 9.803e-12, 8.65e-12, 7.54e-12, 6.619e-12, 5.832e-12,
02052
             5.113e-12, 4.503e-12, 3.975e-12, 3.52e-12, 3.112e-12, 2.797e-12,
02053
02054
             2.5e-12, 2.24e-12, 2.013e-12, 1.819e-12, 1.653e-12, 1.513e-12,
             1.395e-12, 1.299e-12, 1.225e-12, 1.168e-12, 1.124e-12, 1.148e-12,
02055
             1.107e-12, 1.128e-12, 1.169e-12, 1.233e-12, 1.307e-12, 1.359e-12, 1.543e-12, 1.686e-12, 1.794e-12, 2.028e-12, 2.21e-12, 2.441e-12,
02056
02057
             2.653e-12, 2.828e-12, 3.093e-12, 3.28e-12, 3.551e-12, 3.677e-12,
02058
             3.803e-12, 3.844e-12, 4.068e-12, 4.093e-12, 4.002e-12, 3.904e-12,
02059
02060
             3.624e-12, 3.633e-12, 3.622e-12, 3.443e-12, 3.184e-12, 2.934e-12,
            2.476e-12, 2.212e-12, 1.867e-12, 1.594e-12, 1.37e-12, 1.192e-12, 1.045e-12, 9.211e-13, 8.17e-13, 7.29e-13, 6.55e-13, 5.929e-13, 5.415e-13, 4.995e-13, 4.661e-13, 4.406e-13, 4.225e-13, 4.116e-13,
02061
02062
02063
02064
             4.075e-13, 4.102e-13, 4.198e-13, 4.365e-13, 4.606e-13, 4.925e-13,
02065
             5.326e-13, 5.818e-13, 6.407e-13, 7.104e-13, 7.92e-13, 8.868e-13,
02066
             9.964e-13, 1.123e-12, 1.268e-12, 1.434e-12, 1.626e-12, 1.848e-12,
             2.107e-12, 2.422e-12, 2.772e-12, 3.145e-12, 3.704e-12, 4.27e-12, 4.721e-12, 5.361e-12, 6.083e-12, 7.095e-12, 7.968e-12, 9.228e-12,
02067
02068
             1.048e-11, 1.187e-11, 1.336e-11, 1.577e-11, 1.772e-11, 2.017e-11,
02069
             2.25e-11, 2.63e-11, 2.911e-11, 3.356e-11, 3.82e-11, 4.173e-11,
02071
             4.811e-11, 5.254e-11, 5.839e-11, 6.187e-11, 6.805e-11, 7.118e-11,
02072
             7.369e-11, 7.664e-11, 7.794e-11, 7.947e-11, 8.036e-11, 7.954e-11,
02073
             7.849e-11, 7.518e-11, 7.462e-11, 6.926e-11, 6.531e-11, 6.197e-11,
            5.421e-11, 4.777e-11, 4.111e-11, 3.679e-11, 3.166e-11, 2.786e-11, 2.436e-11, 2.144e-11, 1.859e-11, 1.628e-11, 1.414e-11, 1.237e-11,
02074
02075
02076
             1.093e-11, 9.558e-12
02077
02078
02079
          static double h2o260[2001] = { .2752, .2732, .2749, .2676, .2667, .2545,
            .2497, .2327, .2218, .2036, .1825, .1694, .1497, .1353, .121, .1014, .09405, .07848, .07195, .06246, .05306, .04853, .04138, .03735, .03171, .02785, .02431, .02111, .01845, .0164, .01405, .01255, .01098, .009797, .008646, .007779, .006898, .006099, .005453, .004909, .004413, .003959, .003581, .003199, .002871, .002583, .00233, .002086, .001874, .001684, .001512, .001281
02080
02081
02082
02084
02085
             .002583, .00233, .002086, .001874, .001684, .001512, .001361,
             .001225, .0011, 9.89e-4, 8.916e-4, 8.039e-4, 7.256e-4, 6.545e-4, 5.918e-4, 5.359e-4, 4.867e-4, 4.426e-4, 4.033e-4, 3.682e-4, 3.366e-4, 3.085e-4, 2.833e-4, 2.605e-4, 2.403e-4, 2.221e-4,
02086
02087
02088
             2.055e-4, 1.908e-4, 1.774e-4, 1.653e-4, 1.544e-4, 1.443e-4,
             1.351e-4, 1.267e-4, 1.19e-4, 1.119e-4, 1.053e-4, 9.922e-5,
02090
             9.355e-5, 8.831e-5, 8.339e-5, 7.878e-5, 7.449e-5, 7.043e-5, 6.664e-5, 6.307e-5, 5.969e-5, 5.654e-5, 5.357e-5, 5.075e-5, 4.81e-5, 4.56e-5, 4.322e-5, 4.102e-5, 3.892e-5, 3.696e-5,
02091
02092
02093
             3.511e-5, 3.339e-5, 3.177e-5, 3.026e-5, 2.886e-5, 2.756e-5, 2.636e-5, 2.527e-5, 2.427e-5, 2.337e-5, 2.257e-5, 2.185e-5,
02094
             2.127e-5, 2.08e-5, 2.041e-5, 2.013e-5, 2e-5, 1.997e-5, 2.009e-5,
02096
02097
             2.031e-5, 2.068e-5, 2.124e-5, 2.189e-5, 2.267e-5, 2.364e-5,
02098
             2.463e-5, 2.618e-5, 2.774e-5, 2.937e-5, 3.144e-5, 3.359e-5,
02099
             3.695e-5, 4.002e-5, 4.374e-5, 4.947e-5, 5.431e-5, 6.281e-5,
             7.169e-5, 8.157e-5, 9.728e-5, 1.079e-4, 1.337e-4, 1.442e-4,
02100
02101
             1.683e-4, 1.879e-4, 2.223e-4, 2.425e-4, 2.838e-4, 3.143e-4,
             3.527e-4, 4.012e-4, 4.237e-4, 4.747e-4, 5.057e-4, 5.409e-4,
             5.734e-4, 5.944e-4, 6.077e-4, 6.175e-4, 6.238e-4, 6.226e-4,
02103
02104
             6.248e-4, 6.192e-4, 6.098e-4, 5.818e-4, 5.709e-4, 5.465e-4,
02105
             5.043e-4, 4.699e-4, 4.294e-4, 3.984e-4, 3.672e-4, 3.152e-4,
02106
             2.883e-4, 2.503e-4, 2.211e-4, 1.92e-4, 1.714e-4, 1.485e-4,
             1.358e-4, 1.156e-4, 1.021e-4, 8.887e-5, 7.842e-5, 7.12e-5,
02107
             6.186e-5, 5.73e-5, 4.792e-5, 4.364e-5, 3.72e-5, 3.28e-5,
02108
             2.946e-5, 2.591e-5, 2.261e-5, 2.048e-5, 1.813e-5, 1.63e-5,
02109
02110
             1.447e-5, 1.282e-5, 1.167e-5, 1.041e-5, 9.449e-6, 8.51e-6,
02111
             7.596e-6, 6.961e-6, 6.272e-6, 5.728e-6, 5.198e-6, 4.667e-6,
             4.288e-6, 3.897e-6, 3.551e-6, 3.235e-6, 2.952e-6, 2.688e-6,
02112
             2.449e-6, 2.241e-6, 2.05e-6, 1.879e-6, 1.722e-6, 1.582e-6, 1.456e-6, 1.339e-6, 1.236e-6, 1.144e-6, 1.06e-6, 9.83e-7,
02113
             9.149e-7, 8.535e-7, 7.973e-7, 7.466e-7, 6.999e-7, 6.574e-7,
02115
02116
             6.18e-7, 5.821e-7, 5.487e-7, 5.18e-7, 4.896e-7, 4.631e-7,
             4.386e-7, 4.16e-7, 3.945e-7, 3.748e-7, 3.562e-7, 3.385e-7, 3.222e-7, 3.068e-7, 2.922e-7, 2.788e-7, 2.659e-7, 2.539e-7, 2.425e-7, 2.318e-7, 2.219e-7, 2.127e-7, 2.039e-7, 1.958e-7,
02117
02118
02119
```

```
1.885e-7, 1.818e-7, 1.758e-7, 1.711e-7, 1.662e-7, 1.63e-7,
                   1.605e-7, 1.58e-7, 1.559e-7, 1.545e-7, 1.532e-7, 1.522e-7, 1.51e-7, 1.495e-7, 1.465e-7, 1.483e-7, 1.469e-7, 1.448e-7,
02121
02122
                   1.444e-7, 1.436e-7, 1.426e-7, 1.431e-7, 1.425e-7, 1.445e-7,
02123
                   1.477e-7, 1.515e-7, 1.567e-7, 1.634e-7, 1.712e-7, 1.802e-7, 1.914e-7, 2.024e-7, 2.159e-7, 2.295e-7, 2.461e-7, 2.621e-7,
02124
02125
                   2.868e-7, 3.102e-7, 3.394e-7, 3.784e-7, 4.223e-7, 4.864e-7,
                   5.501e-7, 6.039e-7, 7.193e-7, 7.728e-7, 9.514e-7, 1.073e-6,
02127
                   1.18e-6, 1.333e-6, 1.472e-6, 1.566e-6, 1.677e-6, 1.784e-6, 1.904e-6, 1.953e-6, 2.02e-6, 2.074e-6, 2.128e-6, 2.162e-6,
02128
02129
                   2.219e-6, 2.221e-6, 2.249e-6, 2.239e-6, 2.235e-6, 2.185e-6,
02130
                   2.141e-6, 2.124e-6, 2.09e-6, 2.068e-6, 2.1e-6, 2.104e-6,
02131
                   2.142e-6, 2.181e-6, 2.257e-6, 2.362e-6, 2.5e-6, 2.664e-6,
02132
                   2.884e-6, 3.189e-6, 3.48e-6, 3.847e-6, 4.313e-6, 4.79e-6, 5.25e-6, 5.989e-6, 6.692e-6, 7.668e-6, 8.52e-6, 9.606e-6,
02133
02134
                   1.073e-5, 1.225e-5, 1.377e-5, 1.582e-5, 1.761e-5, 2.029e-5, 2.284e-5, 2.602e-5, 2.94e-5, 3.483e-5, 3.928e-5, 4.618e-5, 5.24e-5, 6.132e-5, 7.183e-5, 8.521e-5, 9.111e-5, 1.07e-4,
02135
02136
02137
                   1.184e-4, 1.264e-4, 1.475e-4, 1.612e-4, 1.704e-4, 1.818e-4
                   1.924e-4, 1.994e-4, 2.061e-4, 2.18e-4, 2.187e-4, 2.2e-4,
02139
                   2.196e-4, 2.131e-4, 2.015e-4, 1.988e-4, 1.847e-4, 1.729e-4,
02140
02141
                   1.597e-4, 1.373e-4, 1.262e-4, 1.087e-4, 9.439e-5, 8.061e-5,
                   7.093e-5, 6.049e-5, 5.12e-5, 4.435e-5, 3.817e-5, 3.34e-5,
02142
                  2.927e-5, 2.573e-5, 2.291e-5, 2.04e-5, 1.827e-5, 1.636e-5, 1.463e-5, 1.309e-5, 1.17e-5, 1.047e-5, 9.315e-6, 8.328e-6,
02143
02144
                   7.458e-6, 6.665e-6, 5.94e-6, 5.316e-6, 4.752e-6, 4.252e-6,
                   3.825e-6, 3.421e-6, 3.064e-6, 2.746e-6, 2.465e-6, 2.216e-6,
02146
                  3.825e-6, 3.421e-6, 3.064e-6, 2.746e-6, 2.455e-6, 2.216e-6, 1.99e-6, 1.79e-6, 1.609e-6, 1.449e-6, 1.306e-6, 1.177e-6, 1.063e-6, 9.607e-7, 8.672e-7, 7.855e-7, 7.118e-7, 6.46e-7, 5.871e-7, 5.34e-7, 4.868e-7, 4.447e-7, 4.068e-7, 3.729e-7, 3.423e-7, 3.151e-7, 2.905e-7, 2.686e-7, 2.484e-7, 2.306e-7, 2.142e-7, 1.995e-7, 1.86e-7, 1.738e-7, 1.626e-7, 1.522e-7, 1.427e-7, 1.338e-7, 1.258e-7, 1.183e-7, 1.116e-7, 1.056e-7, 1
02147
02148
02149
02150
02151
02152
                   9.972e-8, 9.46e-8, 9.007e-8, 8.592e-8, 8.195e-8, 7.816e-8,
02153
                  7.483e-8, 7.193e-8, 6.892e-8, 6.642e-8, 6.386e-8, 6.154e-8, 5.949e-8, 5.764e-8, 5.622e-8, 5.479e-8, 5.364e-8, 5.301e-8, 5.267e-8, 5.263e-8, 5.313e-8, 5.41e-8, 5.55e-8, 5.745e-8, 6.003e-8, 6.311e-8, 6.713e-8, 7.173e-8, 7.724e-8, 8.368e-8,
02154
02155
02156
02158
                   9.121e-8, 9.986e-8, 1.097e-7, 1.209e-7, 1.338e-7, 1.486e-7,
                   1.651e-7, 1.837e-7, 2.048e-7, 2.289e-7, 2.557e-7, 2.857e-7,
02159
02160
                   3.195e-7, 3.587e-7, 4.015e-7, 4.497e-7, 5.049e-7, 5.665e-7,
                   6.366e-7, 7.121e-7, 7.996e-7, 8.946e-7, 1.002e-6, 1.117e-6,
02161
                   1.262e-6, 1.416e-6, 1.611e-6, 1.807e-6, 2.056e-6, 2.351e-6, 2.769e-6, 3.138e-6, 3.699e-6, 4.386e-6, 5.041e-6, 6.074e-6,
02162
02163
                   6.812e-6, 7.79e-6, 8.855e-6, 1.014e-5, 1.095e-5, 1.245e-5, 1.316e-5, 1.39e-5, 1.504e-5, 1.583e-5, 1.617e-5, 1.652e-5,
02164
02165
02166
                   1.713e-5, 1.724e-5, 1.715e-5, 1.668e-5, 1.629e-5, 1.552e-5,
                   1.478e-5, 1.34e-5, 1.245e-5, 1.121e-5, 9.575e-6, 8.956e-6, 7.345e-6, 6.597e-6, 5.612e-6, 4.818e-6, 4.165e-6, 3.579e-6,
02167
02168
                   3.041e-6, 2.623e-6, 2.29e-6, 1.984e-6, 1.748e-6, 1.534e-6, 1.369e-6, 1.219e-6, 1.092e-6, 9.8e-7, 8.762e-7, 7.896e-7,
02169
                   7.104e-7, 6.364e-7, 5.691e-7, 5.107e-7, 4.575e-7, 4.09e-7, 3.667e-7, 3.287e-7, 2.931e-7, 2.633e-7, 2.356e-7, 2.111e-7,
02171
02172
02173
                   1.895e-7, 1.697e-7, 1.525e-7, 1.369e-7, 1.233e-7, 1.114e-7,
                   9.988e-8, 9.004e-8, 8.149e-8, 7.352e-8, 6.662e-8, 6.03e-8,
02174
02175
                   5.479e-8, 4.974e-8, 4.532e-8, 4.129e-8, 3.781e-8, 3.462e-8,
                   3.176e-8, 2.919e-8, 2.687e-8, 2.481e-8, 2.292e-8, 2.119e-8,
                   1.967e-8, 1.828e-8, 1.706e-8, 1.589e-8, 1.487e-8, 1.393e-8,
02177
02178
                   1.307e-8, 1.228e-8, 1.156e-8, 1.089e-8, 1.028e-8, 9.696e-9
                   9.159e-9, 8.658e-9, 8.187e-9, 7.746e-9, 7.34e-9, 6.953e-9, 6.594e-9, 6.259e-9, 5.948e-9, 5.66e-9, 5.386e-9, 5.135e-9,
02179
02180
                   4.903e-9, 4.703e-9, 4.515e-9, 4.362e-9, 4.233e-9, 4.117e-9,
4.017e-9, 3.962e-9, 3.924e-9, 3.905e-9, 3.922e-9, 3.967e-9,
02181
02182
                   4.046e-9, 4.165e-9, 4.32e-9, 4.522e-9, 4.769e-9, 5.083e-9, 5.443e-9, 5.872e-9, 6.366e-9, 6.949e-9, 7.601e-9, 8.371e-9,
02183
02184
02185
                   9.22e-9, 1.02e-8, 1.129e-8, 1.251e-8, 1.393e-8, 1.542e-8,
02186
                   1.72e-8, 1.926e-8, 2.152e-8, 2.392e-8, 2.678e-8, 3.028e-8, 3.39e-8, 3.836e-8, 4.309e-8, 4.9e-8, 5.481e-8, 6.252e-8,
02187
                   7.039e-8, 7.883e-8, 8.849e-8, 1.012e-7, 1.142e-7, 1.3e-7,
02188
                   1.475e-7, 1.732e-7, 1.978e-7, 2.304e-7, 2.631e-7, 2.988e-
                   3.392e-7, 3.69e-7, 4.355e-7, 4.672e-7, 5.11e-7, 5.461e-7, 5.828e-7, 6.233e-7, 6.509e-7, 6.672e-7, 6.969e-7, 7.104e-7,
02190
02191
                   7.439e-7, 7.463e-7, 7.708e-7, 7.466e-7, 7.668e-7, 7.549e-7, 7.586e-7, 7.384e-7, 7.439e-7, 7.785e-7, 7.915e-7, 8.31e-7,
02192
02193
                   8.745e-7, 9.558e-7, 1.038e-6, 1.173e-6, 1.304e-6, 1.452e-6, 1.671e-6, 1.931e-6, 2.239e-6, 2.578e-6, 3.032e-6, 3.334e-6,
02194
02195
                   3.98e-6, 4.3e-6, 4.518e-6, 5.321e-6, 5.508e-6, 6.211e-6, 6.59e-6,
02196
                   7.046e-6, 7.555e-6, 7.558e-6, 7.875e-6, 8.319e-6, 8.433e-6, 8.59e-6, 8.503e-6, 8.304e-6, 8.336e-6, 7.739e-6, 7.301e-6,
02197
02198
                   6.827e-6, 6.078e-6, 5.551e-6, 4.762e-6, 4.224e-6, 3.538e-6,
02199
                   2.984e-6, 2.619e-6, 2.227e-6, 1.923e-6, 1.669e-6, 1.462e-6,
02200
                   1.294e-6, 1.155e-6, 1.033e-6, 9.231e-7, 8.238e-7, 7.36e-7,
                   6.564e-7, 5.869e-7, 5.236e-7, 4.673e-7, 4.174e-7, 3.736e-7, 3.33e-7, 2.976e-7, 2.657e-7, 2.367e-7, 2.106e-7, 1.877e-7,
02202
02203
                   1.671e-7, 1.494e-7, 1.332e-7, 1.192e-7, 1.065e-7, 9.558e-8,
02204
                   8.586e-8, 7.717e-8, 6.958e-8, 6.278e-8, 5.666e-8, 5.121e-8, 4.647e-8, 4.213e-8, 3.815e-8, 3.459e-8, 3.146e-8, 2.862e-8,
02205
02206
```

```
2.604e-8, 2.375e-8, 2.162e-8, 1.981e-8, 1.817e-8, 1.67e-8,
             1.537e-8, 1.417e-8, 1.31e-8, 1.215e-8, 1.128e-8, 1.05e-8, 9.793e-9, 9.158e-9, 8.586e-9, 8.068e-9, 7.595e-9, 7.166e-9,
02208
02209
             6.778e-9, 6.427e-9, 6.108e-9, 5.826e-9, 5.571e-9, 5.347e-9,
02210
             5.144e-9, 4.968e-9, 4.822e-9, 4.692e-9, 4.589e-9, 4.506e-9, 4.467e-9, 4.44e-9, 4.466e-9, 4.515e-9, 4.718e-9, 4.729e-9,
02211
02212
             4.937e-9, 5.249e-9, 5.466e-9, 5.713e-9, 6.03e-9, 6.436e-9,
02213
             6.741e-9, 7.33e-9, 7.787e-9, 8.414e-9, 8.908e-9, 9.868e-9,
02214
02215
             1.069e-8, 1.158e-8, 1.253e-8, 1.3e-8, 1.409e-8, 1.47e-8,
02216
             1.548e-8, 1.612e-8, 1.666e-8, 1.736e-8, 1.763e-8, 1.812e-8,
             1.852e-8, 1.923e-8, 1.897e-8, 1.893e-8, 1.888e-8, 1.868e-8,
02217
             1.895e-8, 1.899e-8, 1.876e-8, 1.96e-8, 2.02e-8, 2.121e-8,
02218
02219
             2.239e-8, 2.379e-8, 2.526e-8, 2.766e-8, 2.994e-8, 3.332e-8,
             3.703e-8, 4.158e-8, 4.774e-8, 5.499e-8, 6.355e-8, 7.349e-8,
02220
02221
             8.414e-8, 9.846e-8, 1.143e-7, 1.307e-7, 1.562e-7, 1.817e-7,
            2.011e-7, 2.192e-7, 2.485e-7, 2.867e-7, 3.035e-7, 3.223e-7, 3.443e-7, 3.617e-7, 3.793e-7, 3.793e-7, 3.839e-7, 4.081e-7, 4.117e-7, 4.085e-7, 3.92e-7, 3.851e-7, 3.754e-7, 3.49e-7, 3.229e-7, 2.978e-7, 2.691e-7, 2.312e-7, 2.029e-7, 1.721e-7,
02222
02223
02224
             1.472e-7, 1.308e-7, 1.132e-7, 9.736e-8, 8.458e-8, 7.402e-8,
02226
             6.534e-8, 5.811e-8, 5.235e-8, 4.762e-8, 4.293e-8, 3.896e-8,
02227
02228
             3.526e-8, 3.165e-8, 2.833e-8, 2.551e-8, 2.288e-8, 2.036e-8,
             1.82e-8, 1.626e-8, 1.438e-8, 1.299e-8, 1.149e-8, 1.03e-8, 9.148e-9, 8.122e-9, 7.264e-9, 6.425e-9, 5.777e-9, 5.06e-9, 4.502e-9, 4.013e-9, 3.567e-9, 3.145e-9, 2.864e-9, 2.553e-9,
02229
02230
02231
02232
             2.311e-9, 2.087e-9, 1.886e-9, 1.716e-9, 1.556e-9, 1.432e-9,
             1.311e-9, 1.202e-9, 1.104e-9, 1.013e-9, 9.293e-10, 8.493e-10,
02233
02234
             7.79e-10, 7.185e-10, 6.642e-10, 6.141e-10, 5.684e-10, 5.346e-10,
            5.032e-10, 4.725e-10, 4.439e-10, 4.176e-10, 3.93e-10, 3.714e-10, 3.515e-10, 3.332e-10, 3.167e-10, 3.02e-10, 2.887e-10, 2.769e-10,
02235
02236
             2.665e-10, 2.578e-10, 2.503e-10, 2.436e-10, 2.377e-10, 2.342e-10, 2.305e-10, 2.296e-10, 2.278e-10, 2.321e-10, 2.355e-10, 2.402e-10,
02237
             2.478e-10, 2.67e-10, 2.848e-10, 2.982e-10, 3.263e-10, 3.438e-10,
02239
02240
             3.649e-10, 3.829e-10, 4.115e-10, 4.264e-10, 4.473e-10, 4.63e-10,
02241
             4.808e-10, 4.995e-10, 5.142e-10, 5.313e-10, 5.318e-10, 5.358e-10,
             5.452e-10, 5.507e-10, 5.698e-10, 5.782e-10, 5.983e-10, 6.164e-10,
02242
             6.532e-10, 6.811e-10, 7.624e-10, 8.302e-10, 9.067e-10, 9.937e-10,
02243
             1.104e-9, 1.221e-9, 1.361e-9, 1.516e-9, 1.675e-9, 1.883e-9,
02245
             2.101e-9, 2.349e-9, 2.614e-9, 2.92e-9, 3.305e-9, 3.724e-9,
             4.142e-9, 4.887e-9, 5.614e-9, 6.506e-9, 7.463e-9, 8.817e-9,
02246
             9.849e-9, 1.187e-8, 1.321e-8, 1.474e-8, 1.698e-8, 1.794e-8,
02247
             2.09e-8, 2.211e-8, 2.362e-8, 2.556e-8, 2.729e-8, 2.88e-8, 3.046e-8, 3.167e-8, 3.367e-8, 3.457e-8, 3.59e-8, 3.711e-8, 3.826e-8, 4.001e-8, 4.211e-8, 4.315e-8, 4.661e-8, 5.01e-8,
02248
02249
             5.249e-8, 5.84e-8, 6.628e-8, 7.512e-8, 8.253e-8, 9.722e-8, 1.067e-7, 1.153e-7, 1.347e-7, 1.428e-7, 1.577e-7, 1.694e-7,
02251
02252
02253
             1.833e-7, 1.938e-7, 2.108e-7, 2.059e-7, 2.157e-7, 2.185e-7,
            2.208e-7, 2.182e-7, 2.093e-7, 2.014e-7, 1.962e-7, 1.819e-7, 1.713e-7, 1.51e-7, 1.34e-7, 1.154e-7, 9.89e-8, 8.88e-8, 7.673e-8,
02254
02255
             6.599e-8, 5.73e-8, 5.081e-8, 4.567e-8, 4.147e-8, 3.773e-8, 3.46e-8, 3.194e-8, 2.953e-8, 2.759e-8, 2.594e-8, 2.442e-8,
02256
             2.355e-8, 2.283e-8, 2.279e-8, 2.231e-8, 2.279e-8, 2.239e-8
02258
02259
             2.21e-8, 2.309e-8, 2.293e-8, 2.352e-8, 2.415e-8, 2.43e-8,
             2.426e-8, 2.465e-8, 2.5e-8, 2.496e-8, 2.465e-8, 2.445e-8, 2.383e-8, 2.299e-8, 2.165e-8, 2.113e-8, 1.968e-8, 1.819e-8, 1.644e-8, 1.427e-8, 1.27e-8, 1.082e-8, 9.428e-9, 8.091e-9,
02260
02261
02262
             6.958e-9, 5.988e-9, 5.246e-9, 4.601e-9, 4.098e-9, 3.664e-9,
             3.287e-9, 2.942e-9, 2.656e-9, 2.364e-9, 2.118e-9, 1.903e-9,
02264
02265
             1.703e-9, 1.525e-9, 1.365e-9, 1.229e-9, 1.107e-9, 9.96e-10,
02266
             8.945e-10, 8.08e-10, 7.308e-10, 6.616e-10, 5.994e-10, 5.422e-10,
             4.929e-10, 4.478e-10, 4.07e-10, 3.707e-10, 3.379e-10, 3.087e-10,
02267
02268
             2.823e-10, 2.592e-10, 2.385e-10, 2.201e-10, 2.038e-10, 1.897e-10,
             1.774e-10, 1.667e-10, 1.577e-10, 1.502e-10, 1.437e-10, 1.394e-10,
             1.358e-10, 1.324e-10, 1.329e-10, 1.324e-10, 1.36e-10, 1.39e-10,
02270
02271
             1.424e-10, 1.544e-10, 1.651e-10, 1.817e-10, 1.984e-10, 2.195e-10,
            2.438e-10, 2.7e-10, 2.991e-10, 3.322e-10, 3.632e-10, 3.957e-10, 4.36e-10, 4.701e-10, 5.03e-10, 5.381e-10, 5.793e-10, 6.19e-10, 6.596e-10, 7.004e-10, 7.561e-10, 7.934e-10, 8.552e-10, 9.142e-10,
02272
02273
02274
             9.57e-10, 1.027e-9, 1.097e-9, 1.193e-9, 1.334e-9, 1.47e-9,
             1.636e-9, 1.871e-9, 2.122e-9, 2.519e-9, 2.806e-9, 3.203e-9,
             3.846e-9, 4.362e-9, 5.114e-9, 5.643e-9, 6.305e-9, 6.981e-9,
02277
02278
             7.983e-9, 8.783e-9, 9.419e-9, 1.017e-8, 1.063e-8, 1.121e-8,
             1.13e-8, 1.201e-8, 1.225e-8, 1.232e-8, 1.223e-8, 1.177e-8, 1.151e-8, 1.116e-8, 1.047e-8, 9.698e-9, 8.734e-9, 8.202e-9, 7.041e-9, 6.074e-9, 5.172e-9, 4.468e-9, 3.913e-9, 3.414e-9,
02279
02280
02281
             2.975e-9, 2.65e-9, 2.406e-9, 2.173e-9, 2.009e-9, 1.861e-9,
02282
             1.727e-9, 1.612e-9, 1.514e-9, 1.43e-9, 1.362e-9, 1.333e-9,
02283
02284
             1.288e-9, 1.249e-9, 1.238e-9, 1.228e-9, 1.217e-9, 1.202e-9,
02285
             1.209e-9, 1.177e-9, 1.157e-9, 1.165e-9, 1.142e-9, 1.131e-9,
             1.138e-9. 1.117e-9. 1.1e-9. 1.069e-9. 1.023e-9. 1.005e-9.
02286
             9.159e-10, 8.863e-10, 7.865e-10, 7.153e-10, 6.247e-10, 5.846e-10, 5.133e-10, 4.36e-10, 3.789e-10, 3.335e-10, 2.833e-10, 2.483e-10,
02287
             2.155e-10, 1.918e-10, 1.709e-10, 1.529e-10, 1.374e-10, 1.235e-10,
02289
02290
             1.108e-10, 9.933e-11, 8.932e-11, 8.022e-11, 7.224e-11, 6.52e-11,
02291
             5.896e-11, 5.328e-11, 4.813e-11, 4.365e-11, 3.961e-11, 3.594e-11,
             3.266e-11, 2.967e-11, 2.701e-11, 2.464e-11, 2.248e-11, 2.054e-11, 1.878e-11, 1.721e-11, 1.579e-11, 1.453e-11, 1.341e-11, 1.241e-11,
02292
02293
```

```
1.154e-11, 1.078e-11, 1.014e-11, 9.601e-12, 9.167e-12, 8.838e-12,
            8.614e-12, 8.493e-12, 8.481e-12, 8.581e-12, 8.795e-12, 9.131e-12,
02295
02296
            9.601e-12, 1.021e-11, 1.097e-11, 1.191e-11, 1.303e-11, 1.439e-11,
02297
            1.601e-11, 1.778e-11, 1.984e-11, 2.234e-11, 2.474e-11, 2.766e-11,
02298
            3.085e-11, 3.415e-11, 3.821e-11, 4.261e-11, 4.748e-11, 5.323e-11, 5.935e-11, 6.619e-11, 7.418e-11, 8.294e-11, 9.26e-11, 1.039e-10,
02299
            1.156e-10, 1.297e-10, 1.46e-10, 1.641e-10, 1.858e-10, 2.1e-10,
            2.383e-10, 2.724e-10, 3.116e-10, 3.538e-10, 4.173e-10, 4.727e-10,
02301
            5.503e-10, 6.337e-10, 7.32e-10, 8.298e-10, 9.328e-10, 1.059e-9, 1.176e-9, 1.328e-9, 1.445e-9, 1.593e-9, 1.77e-9, 1.954e-9,
02302
02303
            2.175e-9, 2.405e-9, 2.622e-9, 2.906e-9, 3.294e-9, 3.713e-9,
02304
            3.98e-9, 4.384e-9, 4.987e-9, 5.311e-9, 5.874e-9, 6.337e-9, 7.027e-9, 7.39e-9, 7.769e-9, 8.374e-9, 8.605e-9, 9.165e-9,
02305
02306
            9.415e-9, 9.511e-9, 9.704e-9, 9.588e-9, 9.45e-9, 9.086e-9,
02307
02308
            8.798e-9, 8.469e-9, 7.697e-9, 7.168e-9, 6.255e-9, 5.772e-9,
            4.97e-9, 4.27le-9, 3.653e-9, 3.154e-9, 2.742e-9, 2.435e-9, 2.166e-9, 1.936e-9, 1.73le-9, 1.556e-9, 1.399e-9, 1.272e-9,
02309
02310
            1.157e-9, 1.066e-9, 9.844e-10, 9.258e-10, 8.787e-10, 8.421e-10,
02311
            8.083e-10, 8.046e-10, 8.067e-10, 8.181e-10, 8.325e-10, 8.517e-10,
02313
            9.151e-10, 9.351e-10, 9.677e-10, 1.071e-9, 1.126e-9, 1.219e-9,
            1.297e-9, 1.408e-9, 1.476e-9, 1.517e-9, 1.6e-9, 1.649e-9,
02314
02315
            1.678e-9, 1.746e-9, 1.742e-9, 1.728e-9, 1.699e-9, 1.655e-9,
            1.561e-9, 1.48e-9, 1.451e-9, 1.411e-9, 1.171e-9, 1.106e-9, 9.714e-10, 8.523e-10, 7.346e-10, 6.241e-10, 5.371e-10, 4.704e-10, 4.144e-10, 3.683e-10, 3.292e-10, 2.942e-10, 2.62e-10, 2.341e-10,
02316
02317
02318
            2.104e-10, 1.884e-10, 1.7e-10, 1.546e-10, 1.394e-10, 1.265e-10,
            1.14e-10, 1.019e-10, 9.279e-11, 8.283e-11, 7.458e-11, 6.668e-11,
02320
02321
            5.976e-11, 5.33e-11, 4.794e-11, 4.289e-11, 3.841e-11, 3.467e-11,
02322
            3.13e-11, 2.832e-11, 2.582e-11, 2.356e-11, 2.152e-11, 1.97e-11,
            1.808e-11, 1.664e-11, 1.539e-11, 1.434e-11, 1.344e-11, 1.269e-11, 1.209e-11, 1.162e-11, 1.129e-11, 1.108e-11, 1.099e-11, 1.103e-11,
02323
02324
02325
            1.119e-11, 1.148e-11, 1.193e-11, 1.252e-11, 1.329e-11, 1.421e-11,
            1.555e-11, 1.685e-11, 1.839e-11, 2.054e-11, 2.317e-11, 2.571e-11,
02326
02327
            2.839 e^{-11}, \ 3.171 e^{-11}, \ 3.49 e^{-11}, \ 3.886 e^{-11}, \ 4.287 e^{-11}, \ 4.645 e^{-11},
            5.047e-11, 5.592e-11, 6.109e-11, 6.628e-11, 7.381e-11, 8.088e-11, 8.966e-11, 1.045e-10, 1.12e-10, 1.287e-10, 1.486e-10, 1.662e-10,
02328
02329
            1.866e-10, 2.133e-10, 2.524e-10, 2.776e-10, 3.204e-10, 3.559e-10,
02330
            4.028e-10, 4.448e-10, 4.882e-10, 5.244e-10, 5.605e-10, 6.018e-10,
02332
            6.328e-10, 6.579e-10, 6.541e-10, 7.024e-10, 7.074e-10, 7.068e-10,
            7.009e-10, 6.698e-10, 6.545e-10, 6.209e-10, 5.834e-10, 5.412e-10,
02333
02334
            5.001e-10, 4.231e-10, 3.727e-10, 3.211e-10, 2.833e-10, 2.447e-10,
02335
            2.097e-10, 1.843e-10, 1.639e-10, 1.449e-10, 1.27e-10, 1.161e-10,
            1.033e-10, 9.282e-11, 8.407e-11, 7.639e-11, 7.023e-11, 6.474e-11, 6.142e-11, 5.76e-11, 5.568e-11, 5.472e-11, 5.39e-11, 5.455e-11,
02336
02337
            5.54e-11, 5.587e-11, 6.23e-11, 6.49e-11, 6.868e-11, 7.382e-11,
02338
02339
            8.022e-11, 8.372e-11, 9.243e-11, 1.004e-10, 1.062e-10, 1.13e-10,
02340
            1.176e-10, 1.244e-10, 1.279e-10, 1.298e-10, 1.302e-10, 1.312e-10,
02341
            1.295e-10, 1.244e-10, 1.211e-10, 1.167e-10, 1.098e-10, 9.927e-11,
            8.854e-11, 8.011e-11, 7.182e-11, 5.923e-11, 5.212e-11, 4.453e-11,
02342
            3.832e-11, 3.371e-11, 2.987e-11, 2.651e-11, 2.354e-11, 2.093e-11,
02343
            1.863e-11, 1.662e-11, 1.486e-11, 1.331e-11, 1.193e-11, 1.071e-11, 9.628e-12, 8.66e-12, 7.801e-12, 7.031e-12, 6.347e-12, 5.733e-12,
02345
02346
            5.182e-12, 4.695e-12, 4.26e-12, 3.874e-12, 3.533e-12, 3.235e-12,
            2.979e-12, 2.76e-12, 2.579e-12, 2.432e-12, 2.321e-12, 2.246e-12, 2.205e-12, 2.196e-12, 2.223e-12, 2.288e-12, 2.387e-12, 2.525e-12,
02347
02348
            2.704e-12, 2.925e-12, 3.191e-12, 3.508e-12, 3.876e-12, 4.303e-12, 4.793e-12, 5.347e-12, 5.978e-12, 6.682e-12, 7.467e-12, 8.34e-12,
02349
            9.293e-12, 1.035e-11, 1.152e-11, 1.285e-11, 1.428e-11, 1.586e-11,
02351
            1.764e-11, 1.972e-11, 2.214e-11, 2.478e-11, 2.776e-11, 3.151e-11,
02352
02353
            3.591 e^{-11},\ 4.103 e^{-11},\ 4.66 e^{-11},\ 5.395 e^{-11},\ 6.306 e^{-11},\ 7.172 e^{-11},
            8.358e-11, 9.67e-11, 1.11e-10, 1.325e-10, 1.494e-10, 1.736e-10, 2.007e-10, 2.296e-10, 2.608e-10, 3.004e-10, 3.361e-10, 3.727e-10, 4.373e-10, 4.838e-10, 5.483e-10, 6.006e-10, 6.535e-10, 6.899e-10,
02354
02355
            7.687e-10, 8.444e-10, 8.798e-10, 9.135e-10, 9.532e-10,
02357
02358
            9.968e-10, 1.006e-9, 9.949e-10, 9.789e-10, 9.564e-10, 9.215e-10,
02359
            8.51e-10, 8.394e-10, 7.707e-10, 7.152e-10, 6.274e-10, 5.598e-10,
02360
            5.028e-10, 4.3e-10, 3.71e-10, 3.245e-10, 2.809e-10, 2.461e-10,
            2.154e-10, 1.91e-10, 1.685e-10, 1.487e-10, 1.313e-10, 1.163e-10, 1.031e-10, 9.172e-11, 8.221e-11, 7.382e-11, 6.693e-11, 6.079e-11,
02361
02362
            5.581e-11, 5.167e-11, 4.811e-11, 4.506e-11, 4.255e-11, 4.083e-11,
            3.949e-11, 3.881e-11, 3.861e-11, 3.858e-11, 3.951e-11, 4.045e-11,
02364
02365
            4.24e-11, 4.487e-11, 4.806e-11, 5.133e-11, 5.518e-11, 5.919e-11,
            6.533e-11, 7.031e-11, 7.762e-11, 8.305e-11, 9.252e-11, 9.727e-11, 1.045e-10, 1.117e-10, 1.2e-10, 1.275e-10, 1.341e-10, 1.362e-10,
02366
02367
            1.438e-10, 1.45e-10, 1.455e-10, 1.455e-10, 1.434e-10, 1.381e-10, 1.301e-10, 1.276e-10, 1.163e-10, 1.089e-10, 9.911e-11, 8.943e-11,
02368
02369
            7.618e-11, 6.424e-11, 5.717e-11, 4.866e-11, 4.257e-11, 3.773e-11,
02370
02371
            3.331e-11, 2.958e-11, 2.629e-11, 2.316e-11, 2.073e-11, 1.841e-11,
            1.635e-11, 1.464e-11, 1.31e-11, 1.16e-11, 1.047e-11, 9.408e-12, 8.414e-12, 7.521e-12, 6.705e-12, 5.993e-12, 5.371e-12, 4.815e-12,
02372
02373
02374
            4.338e-12, 3.921e-12, 3.567e-12, 3.265e-12, 3.01e-12, 2.795e-12,
            2.613e-12, 2.464e-12, 2.346e-12, 2.256e-12, 2.195e-12, 2.165e-12,
            2.166e-12, 2.198e-12, 2.262e-12, 2.364e-12, 2.502e-12, 2.682e-12,
02376
            2.908e-12, 3.187e-12, 3.533e-12, 3.946e-12, 4.418e-12, 5.013e-12,
02377
02378
            5.708e-12, 6.379e-12, 7.43e-12, 8.39e-12, 9.51e-12, 1.078e-11,
            1.259e-11, 1.438e-11, 1.63e-11, 1.814e-11, 2.055e-11, 2.348e-11, 2.664e-11, 2.956e-11, 3.3e-11, 3.677e-11, 4.032e-11, 4.494e-11,
02379
02380
```

```
4.951e-11, 5.452e-11, 6.014e-11, 6.5e-11, 6.915e-11, 7.45e-11,
            7.971e-11, 8.468e-11, 8.726e-11, 8.95e-11, 9.182e-11, 9.509e-11, 9.333e-11, 9.386e-11, 9.457e-11, 9.21e-11, 9.019e-11, 8.68e-11, 8.298e-11, 7.947e-11, 7.46e-11, 7.082e-11, 6.132e-11, 5.855e-11,
02382
02383
02384
            5.073e-11, 4.464e-11, 3.825e-11, 3.375e-11, 2.911e-11, 2.535e-11, 2.16e-11, 1.907e-11, 1.665e-11, 1.463e-11, 1.291e-11, 1.133e-11,
02385
02386
             9.997e-12, 8.836e-12, 7.839e-12, 6.943e-12, 6.254e-12, 5.6e-12,
             5.029e-12, 4.529e-12, 4.102e-12, 3.737e-12, 3.428e-12, 3.169e-12,
02388
02389
            2.959e-12, 2.798e-12, 2.675e-12, 2.582e-12, 2.644e-12, 2.557e-12,
02390
            2.614e-12, 2.717e-12, 2.874e-12, 3.056e-12, 3.187e-12, 3.631e-12,
            3.979e-12, 4.248e-12, 4.817e-12, 5.266e-12, 5.836e-12, 6.365e-12,
02391
             6.807e-12, 7.47e-12, 7.951e-12, 8.636e-12, 8.972e-12, 9.314e-12,
02392
02393
             9.445e-12, 1.003e-11, 1.013e-11, 9.937e-12, 9.729e-12, 9.064e-12,
             9.119e-12, 9.124e-12, 8.704e-12, 8.078e-12, 7.47e-12, 6.329e-12,
02394
02395
            5.674e-12, 4.808e-12, 4.119e-12, 3.554e-12, 3.103e-12, 2.731e-12,
            2.415e-12, 2.15e-12, 1.926e-12, 1.737e-12, 1.578e-12, 1.447e-12, 1.34e-12, 1.255e-12, 1.191e-12, 1.146e-12, 1.121e-12, 1.114e-12,
02396
02397
02398
            1.126e-12, 1.156e-12, 1.207e-12, 1.278e-12, 1.372e-12, 1.49e-12,
             1.633e-12, 1.805e-12, 2.01e-12, 2.249e-12, 2.528e-12, 2.852e-12,
02400
             3.228e-12, 3.658e-12, 4.153e-12, 4.728e-12, 5.394e-12, 6.176e-12,
             7.126e-12, 8.188e-12, 9.328e-12, 1.103e-11, 1.276e-11, 1.417e-11,
02401
02402
            1.615 e^{-11}, \ 1.84 e^{-11}, \ 2.155 e^{-11}, \ 2.429 e^{-11}, \ 2.826 e^{-11}, \ 3.222 e^{-11},
            3.664e-11, 4.14e-11, 4.906e-11, 5.536e-11, 6.327e-11, 7.088e-11,
02403
            8.316e-11, 9.242e-11, 1.07e-10, 1.223e-10, 1.341e-10, 1.553e-10, 1.703e-10, 1.9e-10, 2.022e-10, 2.233e-10, 2.345e-10, 2.438e-10,
02404
02405
             2.546e-10, 2.599e-10, 2.661e-10, 2.703e-10, 2.686e-10, 2.662e-10,
02406
02407
            2.56e-10, 2.552e-10, 2.378e-10, 2.252e-10, 2.146e-10, 1.885e-10,
02408
            1.668e-10, 1.441e-10, 1.295e-10, 1.119e-10, 9.893e-11, 8.687e-11,
02409
             7.678e-11, 6.685e-11, 5.879e-11, 5.127e-11, 4.505e-11, 3.997e-11,
02410
            3.511e-11
02411
02412
02413
          static double h2ofrn[2001] = { .01095, .01126, .01205, .01322, .0143,
02414
            .01506, .01548, .01534, .01486, .01373, .01262, .01134, .01001,
             .008702, .007475, .006481, .00548, .0046, .003833, .00311, .002543, .002049, .00168, .001374, .001046, 8.193e-4, 6.267e-4,
02415
02416
            4.968e-4, 3.924e-4, 2.983e-4, 2.477e-4, 1.997e-4, 1.596e-4, 1.331e-4, 1.061e-4, 8.942e-5, 7.168e-5, 5.887e-5, 4.848e-5,
02417
02419
             3.817e-5, 3.17e-5, 2.579e-5, 2.162e-5, 1.768e-5, 1.49e-5,
             1.231e-5, 1.013e-5, 8.555e-6, 7.328e-6, 6.148e-6, 5.207e-6,
02420
02421
             4.387e-6, 3.741e-6, 3.22e-6, 2.753e-6, 2.346e-6, 1.985e-6,
            1.716e-6, 1.475e-6, 1.286e-6, 1.122e-6, 9.661e-7, 8.284e-7,
02422
            7.057e-7, 6.119e-7, 5.29e-7, 4.571e-7, 3.948e-7, 3.432e-7, 2.983e-7, 2.589e-7, 2.265e-7, 1.976e-7, 1.704e-7, 1.456e-7,
02423
            1.26e-7, 1.101e-7, 9.648e-8, 8.415e-8, 7.34e-8, 6.441e-8, 5.643e-8, 4.94e-8, 4.276e-8, 3.703e-8, 3.227e-8, 2.825e-8,
02426
02427
            2.478e-8, 2.174e-8, 1.898e-8, 1.664e-8, 1.458e-8, 1.278e-8,
            1.126e-8, 9.891e-9, 8.709e-9, 7.652e-9, 6.759e-9, 5.975e-9, 5.31e-9, 4.728e-9, 4.214e-9, 3.792e-9, 3.463e-9, 3.226e-9,
02428
02429
02430
            2.992e-9, 2.813e-9, 2.749e-9, 2.809e-9, 2.913e-9, 3.037e-9,
             3.413e-9, 3.738e-9, 4.189e-9, 4.808e-9,
                                                                 5.978e-9,
             8.071e-9, 9.61e-9, 1.21e-8, 1.5e-8, 1.764e-8, 2.221e-8, 2.898e-8,
02432
02433
            3.948e-8, 5.068e-8, 6.227e-8, 7.898e-8, 1.033e-7, 1.437e-7,
            1.889e-7, 2.589e-7, 3.59e-7, 4.971e-7, 7.156e-7, 9.983e-7, 1.381e-6, 1.929e-6, 2.591e-6, 3.453e-6, 4.57e-6, 5.93e-6,
02434
02435
             7.552e-6, 9.556e-6, 1.183e-5, 1.425e-5, 1.681e-5, 1.978e-5,
02436
             2.335e-5, 2.668e-5, 3.022e-5, 3.371e-5, 3.715e-5, 3.967e-5,
             4.06e-5, 4.01e-5, 3.809e-5, 3.491e-5, 3.155e-5, 2.848e-5,
02438
02439
            2.678e-5, 2.66e-5, 2.811e-5, 3.071e-5, 3.294e-5, 3.459e-5,
02440
            3.569e-5, 3.56e-5, 3.434e-5, 3.186e-5, 2.916e-5, 2.622e-5,
02441
            2.275e-5, 1.918e-5, 1.62e-5, 1.373e-5, 1.182e-5, 1.006e-5,
02442
            8.556e-6, 7.26e-6, 6.107e-6, 5.034e-6, 4.211e-6, 3.426e-6,
             2.865e-6, 2.446e-6, 1.998e-6, 1.628e-6, 1.242e-6, 1.005e-6,
            7.853e-7, 6.21e-7, 5.071e-7, 4.156e-7, 3.548e-7, 2.825e-7, 2.261e-7, 1.916e-7, 1.51e-7, 1.279e-7, 1.059e-7, 9.14e-8,
02444
02445
02446
            7.707e-8, 6.17e-8, 5.311e-8, 4.263e-8, 3.518e-8, 2.961e-8,
02447
            2.457e-8, 2.119e-8, 1.712e-8, 1.439e-8, 1.201e-8, 1.003e-8, 8.564e-9, 7.199e-9, 6.184e-9, 5.206e-9, 4.376e-9, 3.708e-9,
02448
02449
            3.157e-9, 2.725e-9, 2.361e-9, 2.074e-9, 1.797e-9, 1.562e-9,
             1.364e-9, 1.196e-9, 1.042e-9, 8.862e-10, 7.648e-10, 6.544e-10,
02451
             5.609e-10, 4.791e-10, 4.108e-10, 3.531e-10, 3.038e-10, 2.618e-10,
02452
            2.268e-10, 1.969e-10, 1.715e-10, 1.496e-10, 1.308e-10, 1.147e-10,
            1.008e-10, 8.894e-11, 7.885e-11, 7.031e-11, 6.355e-11, 5.854e-11, 5.534e-11, 5.466e-11, 5.725e-11, 6.447e-11, 7.943e-11, 1.038e-10, 1.437e-10, 2.04e-10, 2.901e-10, 4.051e-10, 5.556e-10, 7.314e-10, 9.291e-10, 1.134e-9, 1.321e-9, 1.482e-9, 1.596e-9, 1.669e-9,
02453
02454
02455
02456
            1.715e-9, 1.762e-9, 1.817e-9, 1.828e-9, 1.848e-9, 1.873e-9,
02457
02458
            1.902e-9, 1.894e-9, 1.864e-9, 1.841e-9, 1.797e-9, 1.704e-9, 1.559e-9, 1.382e-9, 1.187e-9, 1.001e-9, 8.468e-10, 7.265e-10,
02459
             6.521e-10, 6.381e-10, 6.66e-10, 7.637e-10, 9.705e-10, 1.368e-9,
02460
            1.856e-9, 2.656e-9, 3.954e-9, 5.96e-9, 8.72e-9, 1.247e-8, 1.781e-8, 2.491e-8, 3.311e-8, 4.272e-8, 5.205e-8, 6.268e-8,
02461
             7.337e-8, 8.277e-8, 9.185e-8, 1.004e-7, 1.091e-7, 1.159e-7,
02463
02464
            1.188e-7, 1.175e-7, 1.124e-7, 1.033e-7, 9.381e-8, 8.501e-8,
            7.956e-8, 7.894e-8, 8.331e-8, 9.102e-8, 9.836e-8, 1.035e-7, 1.064e-7, 1.06e-7, 1.032e-7, 9.808e-8, 9.139e-8, 8.442e-8, 7.641e-8, 6.881e-8, 6.161e-8, 5.404e-8, 4.804e-8, 4.446e-8,
02465
02466
02467
```

```
4.328e-8, 4.259e-8, 4.421e-8, 4.673e-8, 4.985e-8, 5.335e-8,
             5.796e-8, 6.542e-8, 7.714e-8, 8.827e-8, 1.04e-7, 1.238e-7, 1.499e-7, 1.829e-7, 2.222e-7, 2.689e-7, 3.303e-7, 3.981e-7,
02469
02470
             4.84e-7, 5.91e-7, 7.363e-7, 9.087e-7, 1.139e-6, 1.455e-6,
02471
             1.866e-6, 2.44e-6, 3.115e-6, 3.941e-6, 4.891e-6, 5.992e-6, 7.111e-6, 8.296e-6, 9.21e-6, 9.987e-6, 1.044e-5, 1.073e-5,
02472
02473
             1.092e-5, 1.106e-5, 1.138e-5, 1.171e-5, 1.186e-5, 1.186e-5,
             1.179e-5, 1.166e-5, 1.151e-5, 1.16e-5, 1.197e-5, 1.241e-5,
02475
02476
             1.268e-5, 1.26e-5, 1.184e-5, 1.063e-5, 9.204e-6, 7.584e-6,
             6.053e-6, 4.482e-6, 3.252e-6, 2.337e-6, 1.662e-6, 1.18e-6, 8.15e-7, 5.95e-7, 4.354e-7, 3.302e-7, 2.494e-7, 1.93e-7, 1.545e-7, 1.25e-7, 1.039e-7, 8.602e-8, 7.127e-8, 5.897e-8, 4.838e-8, 4.018e-8, 3.28e-8, 2.72e-8, 2.307e-8, 1.972e-8,
02477
02478
02479
02480
             1.654e-8, 1.421e-8, 1.174e-8, 1.004e-8, 8.739e-9, 7.358e-9,
02481
02482
             6.242e-9, 5.303e-9, 4.567e-9, 3.94e-9, 3.375e-9, 2.864e-9,
             2.422e-9, 2.057e-9, 1.75e-9, 1.505e-9, 1.294e-9, 1.101e-9, 9.401e-10, 8.018e-10, 6.903e-10, 5.965e-10, 5.087e-10, 4.364e-10, 3.759e-10, 3.247e-10, 2.809e-10, 2.438e-10, 2.123e-10, 1.853e-10,
02483
02484
02485
             1.622e-10, 1.426e-10, 1.26e-10, 1.125e-10, 1.022e-10, 9.582e-11,
             9.388e-11, 9.801e-11, 1.08e-10, 1.276e-10, 1.551e-10, 1.903e-10,
             2.291e-10, 2.724e-10, 3.117e-10, 3.4e-10, 3.562e-10, 3.625e-10,
02488
02489
             3.619e-10, 3.429e-10, 3.221e-10, 2.943e-10, 2.645e-10, 2.338e-10,
             2.062e-10, 1.901e-10, 1.814e-10, 1.827e-10, 1.906e-10, 1.984e-10, 2.04e-10, 2.068e-10, 2.075e-10, 2.018e-10, 1.959e-10, 1.897e-10, 1.852e-10, 1.791e-10, 1.696e-10, 1.634e-10, 1.598e-10, 1.561e-10,
02490
02491
02492
             1.518e-10, 1.443e-10, 1.377e-10, 1.346e-10, 1.342e-10, 1.375e-10,
             1.525e-10, 1.767e-10, 2.108e-10, 2.524e-10, 2.981e-10,
02494
02/95
             4.262e-10, 5.326e-10, 6.646e-10, 8.321e-10, 1.069e-9, 1.386e-9,
02496
             1.743e-9, 2.216e-9, 2.808e-9, 3.585e-9, 4.552e-9, 5.907e-9,
02497
             7.611e-9, 9.774e-9, 1.255e-8, 1.666e-8, 2.279e-8, 3.221e-8,
             4.531e-8, 6.4e-8, 9.187e-8, 1.295e-7, 1.825e-7, 2.431e-7, 3.181e-7, 4.009e-7, 4.941e-7, 5.88e-7, 6.623e-7, 7.155e-7,
02498
02499
             7.451e-7, 7.594e-7, 7.541e-7, 7.467e-7, 7.527e-7, 7.95e-7, 8.461e-7, 8.954e-7, 9.364e-7, 9.843e-7, 1.024e-6, 1.05e-6,
02500
02501
             1.059e-6, 1.074e-6, 1.072e-6, 1.043e-6, 9.789e-7, 8.803e-7, 7.662e-7, 6.378e-7, 5.133e-7, 3.958e-7, 2.914e-7, 2.144e-7, 1.57e-7, 1.14e-7, 8.47e-8, 6.2e-8, 4.657e-8, 3.559e-8, 2.813e-8,
02502
02503
02504
             2.222e-8, 1.769e-8, 1.391e-8, 1.125e-8, 9.186e-9, 7.704e-9,
02506
             6.447e-9, 5.381e-9, 4.442e-9, 3.669e-9, 3.057e-9, 2.564e-9,
             2.153e-9, 1.784e-9, 1.499e-9, 1.281e-9, 1.082e-9, 9.304e-10
02507
02508
             8.169e-10, 6.856e-10, 5.866e-10, 5.043e-10, 4.336e-10, 3.731e-10,
             3.175e-10, 2.745e-10, 2.374e-10, 2.007e-10, 1.737e-10, 1.508e-10, 1.302e-10, 1.13e-10, 9.672e-11, 8.375e-11, 7.265e-11, 6.244e-11, 5.343e-11, 4.654e-11, 3.975e-11, 3.488e-11, 3.097e-11, 2.834e-11,
02509
02510
02512
             2.649e-11,\ 2.519e-11,\ 2.462e-11,\ 2.443e-11,\ 2.44e-11,\ 2.398e-11,
02513
             2.306e-11, 2.183e-11, 2.021e-11, 1.821e-11, 1.599e-11, 1.403e-11,
02514
             1.196e-11, 1.023e-11, 8.728e-12, 7.606e-12, 6.941e-12, 6.545e-12,
             6.484e-12, 6.6e-12, 6.718e-12, 6.785e-12, 6.746e-12, 6.724e-12, 6.764e-12, 6.995e-12, 7.144e-12, 7.32e-12, 7.33e-12, 7.208e-12, 6.789e-12, 6.09e-12, 5.337e-12, 4.62e-12, 4.037e-12, 3.574e-12,
02515
02516
02517
             3.311e-12, 3.346e-12, 3.566e-12, 3.836e-12, 4.076e-12, 4.351e-12,
             4.691e-12, 5.114e-12, 5.427e-12, 6.167e-12, 7.436e-12, 8.842e-12,
02519
02520
             1.038 e^{-11}, \ 1.249 e^{-11}, \ 1.54 e^{-11}, \ 1.915 e^{-11}, \ 2.48 e^{-11}, \ 3.256 e^{-11},
             4.339e-11, 5.611e-11, 7.519e-11, 1.037e-10, 1.409e-10, 1.883e-10, 2.503e-10, 3.38e-10, 4.468e-10, 5.801e-10, 7.335e-10, 8.98e-10,
02521
02522
             1.11e-9, 1.363e-9, 1.677e-9, 2.104e-9, 2.681e-9, 3.531e-9,
02523
              4.621e-9, 6.106e-9, 8.154e-9, 1.046e-8, 1.312e-8, 1.607e-8,
             1.948e-8, 2.266e-8, 2.495e-8, 2.655e-8, 2.739e-8, 2.739e-8,
02525
02526
             2.662e-8, 2.589e-8, 2.59e-8, 2.664e-8, 2.833e-8, 3.023e-8
02527
             3.305e-8, 3.558e-8, 3.793e-8, 3.961e-8, 4.056e-8, 4.102e-8,
02528
             4.025e-8, 3.917e-8, 3.706e-8, 3.493e-8, 3.249e-8, 3.096e-8,
             3.011e-8, 3.111e-8, 3.395e-8, 3.958e-8, 4.875e-8, 6.066e-8,
02529
             7.915e-8, 1.011e-7, 1.3e-7, 1.622e-7, 2.003e-7, 2.448e-7,
             2.863e-7, 3.317e-7, 3.655e-7, 3.96e-7, 4.098e-7, 4.168e-7, 4.198e-7, 4.207e-7, 4.289e-7, 4.384e-7, 4.471e-7, 4.524e-7,
02531
02532
02533
             4.574e-7, 4.633e-7, 4.785e-7, 5.028e-7, 5.371e-7, 5.727e-7,
02534
             5.955e-7, 5.998e-7, 5.669e-7, 5.082e-7, 4.397e-7, 3.596e-7, 2.814e-7, 2.074e-7, 1.486e-7, 1.057e-7, 7.25e-8, 4.946e-8,
02535
             3.43e-8, 2.447e-8, 1.793e-8, 1.375e-8, 1.096e-8, 9.091e-9,
02536
              7.709e-9, 6.631e-9, 5.714e-9, 4.886e-9, 4.205e-9, 3.575e-9,
             3.07e-9, 2.631e-9, 2.284e-9, 2.002e-9, 1.745e-9, 1.509e-9,
02538
02539
             1.284e-9, 1.084e-9, 9.163e-10, 7.663e-10, 6.346e-10, 5.283e-10,
             4.354e-10, 3.59e-10, 2.982e-10, 2.455e-10, 2.033e-10, 1.696e-10, 1.432e-10, 1.211e-10, 1.02e-10, 8.702e-11, 7.38e-11, 6.293e-11,
02540
02541
             5.343e-11, 4.532e-11, 3.907e-11, 3.365e-11, 2.945e-11, 2.558e-11, 2.192e-11, 1.895e-11, 1.636e-11, 1.42e-11, 1.228e-11, 1.063e-11,
02542
02543
             9.348e-12, 8.2e-12, 7.231e-12, 6.43e-12, 5.702e-12, 5.052e-12,
02544
02545
             4.469e-12, 4e-12, 3.679e-12, 3.387e-12, 3.197e-12, 3.158e-12,
02546
             3.327e-12, 3.675e-12, 4.292e-12, 5.437e-12, 7.197e-12, 1.008e-11,
             1.437e-11,\ 2.035e-11,\ 2.905e-11,\ 4.062e-11,\ 5.528e-11,\ 7.177e-11,
02547
             9.064e-11, 1.109e-10, 1.297e-10, 1.473e-10, 1.652e-10, 1.851e-10, 2.079e-10, 2.313e-10, 2.619e-10, 2.958e-10, 3.352e-10, 3.796e-10,
02548
             4.295e-10, 4.923e-10, 5.49e-10, 5.998e-10, 6.388e-10, 6.645e-10,
02550
02551
             6.712e-10, 6.549e-10, 6.38e-10, 6.255e-10, 6.253e-10, 6.459e-10,
02552
             6.977e-10, 7.59e-10, 8.242e-10, 8.92e-10, 9.403e-10, 9.701e-10,
             9.483e-10, 9.135e-10, 8.617e-10, 7.921e-10, 7.168e-10, 6.382e-10, 5.677e-10, 5.045e-10, 4.572e-10, 4.312e-10, 4.145e-10, 4.192e-10,
02553
02554
```

```
4.541e-10, 5.368e-10, 6.771e-10, 8.962e-10, 1.21e-9, 1.659e-9,
            2.33e-9, 3.249e-9, 4.495e-9, 5.923e-9, 7.642e-9, 9.607e-9, 1.178e-8, 1.399e-8, 1.584e-8, 1.73e-8, 1.816e-8, 1.87e-8,
02556
02557
02558
            1.868e-8, 1.87e-8, 1.884e-8, 1.99e-8, 2.15e-8, 2.258e-8,
            2.364e-8, 2.473e-8, 2.602e-8, 2.689e-8, 2.731e-8, 2.816e-8, 2.859e-8, 2.839e-8, 2.703e-8, 2.451e-8, 2.149e-8, 1.787e-8,
02559
02560
             1.449e-8, 1.111e-8, 8.282e-9, 6.121e-9, 4.494e-9, 3.367e-9,
             2.487e-9, 1.885e-9, 1.503e-9, 1.249e-9, 1.074e-9, 9.427e-10,
02562
02563
             8.439e-10, 7.563e-10, 6.772e-10, 6.002e-10, 5.254e-10, 4.588e-10,
02564
            3.977e-10,\ 3.449e-10,\ 3.003e-10,\ 2.624e-10,\ 2.335e-10,\ 2.04e-10,
            1.771e-10, 1.534e-10, 1.296e-10, 1.097e-10, 9.173e-11, 7.73e-11,
02565
02566
             6.547e-11, 5.191e-11, 4.198e-11, 3.361e-11, 2.732e-11, 2.244e-11,
             1.791e-11, 1.509e-11, 1.243e-11, 1.035e-11, 8.969e-12, 7.394e-12,
02567
02568
             6.323e-12, 5.282e-12, 4.543e-12, 3.752e-12, 3.14e-12, 2.6e-12,
02569
            2.194e-12, 1.825e-12, 1.511e-12, 1.245e-12, 1.024e-12, 8.539e-13,
            7.227e-13, 6.102e-13, 5.189e-13, 4.43e-13, 3.774e-13, 3.236e-13, 2.8e-13, 2.444e-13, 2.156e-13, 1.932e-13, 1.775e-13, 1.695e-13,
02570
02571
            1.672e-13, 1.704e-13, 1.825e-13, 2.087e-13, 2.614e-13, 3.377e-13,
02572
             4.817e-13, 6.989e-13, 1.062e-12, 1.562e-12, 2.288e-12, 3.295e-12,
             4.55e-12, 5.965e-12, 7.546e-12, 9.395e-12, 1.103e-11, 1.228e-11,
02574
            1.318e-11, 1.38e-11, 1.421e-11, 1.39e-11, 1.358e-11, 1.336e-11,
02575
02576
            1.342e-11, 1.356e-11, 1.424e-11, 1.552e-11, 1.73e-11, 1.951e-11,
             2.128e-11, \ 2.249e-11, \ 2.277e-11, \ 2.226e-11, \ 2.111e-11, \ 1.922e-11, \\
02577
            1.775e-11, 1.661e-11, 1.547e-11, 1.446e-11, 1.323e-11, 1.21e-11, 1.054e-11, 9.283e-12, 8.671e-12, 8.67e-12, 9.429e-12, 1.062e-11,
02578
            1.255e-11, 1.506e-11, 1.818e-11, 2.26e-11, 2.831e-11, 3.723e-11, 5.092e-11, 6.968e-11, 9.826e-11, 1.349e-10, 1.87e-10, 2.58e-10,
02580
02581
02582
            3.43e-10, 4.424e-10, 5.521e-10, 6.812e-10, 8.064e-10, 9.109e-10,
            9.839e-10, 1.028e-9, 1.044e-9, 1.029e-9, 1.005e-9, 1.002e-9, 1.038e-9, 1.122e-9, 1.233e-9, 1.372e-9, 1.524e-9, 1.665e-9, 1.804e-9, 1.908e-9, 2.015e-9, 2.117e-9, 2.219e-9, 2.336e-9,
02583
02584
02585
             2.531e-9, 2.805e-9, 3.189e-9, 3.617e-9, 4.208e-9, 4.911e-9,
             5.619e-9, 6.469e-9, 7.188e-9, 7.957e-9, 8.503e-9, 9.028e-9,
02587
02588
            9.571e-9, 9.99e-9, 1.055e-8, 1.102e-8, 1.132e-8, 1.141e-8,
            1.145e-8, 1.145e-8, 1.176e-8, 1.224e-8, 1.304e-8, 1.388e-8, 1.445e-8, 1.453e-8, 1.368e-8, 1.22e-8, 1.042e-8, 8.404e-9,
02589
02590
             6.403e-9, 4.643e-9, 3.325e-9, 2.335e-9, 1.638e-9, 1.19e-9,
02591
             9.161e-10, 7.412e-10, 6.226e-10, 5.516e-10, 5.068e-10, 4.831e-10,
02593
             4.856e-10, 5.162e-10, 5.785e-10, 6.539e-10, 7.485e-10,
02594
             9.534e-10, 1.052e-9, 1.115e-9, 1.173e-9, 1.203e-9, 1.224e-9,
02595
            1.243e-9, 1.248e-9, 1.261e-9, 1.265e-9, 1.25e-9, 1.217e-9,
            1.176e-9, 1.145e-9, 1.153e-9, 1.199e-9, 1.278e-9, 1.366e-9, 1.426e-9, 1.444e-9, 1.365e-9, 1.224e-9, 1.051e-9, 8.539e-10,
02596
02597
             6.564e-10, 4.751e-10, 3.404e-10, 2.377e-10, 1.631e-10, 1.114e-10,
             7.87e-11, 5.793e-11, 4.284e-11, 3.3e-11, 2.62e-11, 2.152e-11,
02599
02600
            1.777e-11, 1.496e-11, 1.242e-11, 1.037e-11, 8.725e-12, 7.004e-12,
02601
            5.718e-12, 4.769e-12, 3.952e-12, 3.336e-12, 2.712e-12, 2.213e-12,
            1.803e-12, 1.492e-12, 1.236e-12, 1.006e-12, 8.384e-13, 7.063e-13, 5.879e-13, 4.93e-13, 4.171e-13, 3.569e-13, 3.083e-13, 2.688e-13, 2.333e-13, 2.035e-13, 1.82e-13, 1.682e-13, 1.635e-13, 1.674e-13,
02602
02603
02604
             1.769e-13, 2.022e-13, 2.485e-13, 3.127e-13, 4.25e-13, 5.928e-13,
             8.514e-13, 1.236e-12, 1.701e-12, 2.392e-12, 3.231e-12, 4.35e-12,
02606
02607
            5.559e-12, 6.915e-12, 8.519e-12, 1.013e-11, 1.146e-11, 1.24e-11,
02608
            1.305e-11, 1.333e-11, 1.318e-11, 1.263e-11, 1.238e-11, 1.244e-11,
02609
            1.305e-11, 1.432e-11, 1.623e-11, 1.846e-11, 2.09e-11, 2.328e-11,
            2.526e-11, 2.637e-11, 2.702e-11, 2.794e-11, 2.889e-11, 2.989e-11, 3.231e-11, 3.68e-11, 4.375e-11, 5.504e-11, 7.159e-11, 9.502e-11,
02610
            1.279e-10, 1.645e-10, 2.098e-10, 2.618e-10, 3.189e-10, 3.79e-10,
02612
02613
             4.303e-10, 4.753e-10, 5.027e-10, 5.221e-10, 5.293e-10, 5.346e-10,
            5.467e-10, 5.796e-10, 6.2e-10, 6.454e-10, 6.705e-10, 6.925e-10, 7.233e-10, 7.35e-10, 7.538e-10, 7.861e-10, 8.077e-10, 8.132e-10,
02614
02615
            7.749e-10, 7.036e-10, 6.143e-10, 5.093e-10, 4.089e-10, 3.092e-10, 2.299e-10, 1.705e-10, 1.277e-10, 9.723e-11, 7.533e-11, 6.126e-11,
02616
            5.154e-11, 4.428e-11, 3.913e-11, 3.521e-11, 3.297e-11, 3.275e-11, 3.46e-11, 3.798e-11, 4.251e-11, 4.745e-11, 5.232e-11, 5.606e-11,
02618
02619
02620
            5.82e-11, 5.88e-11, 5.79e-11, 5.661e-11, 5.491e-11, 5.366e-11,
02621
            5.341e-11, 5.353e-11, 5.336e-11, 5.293e-11, 5.248e-11, 5.235e-11,
             5.208e-11, 5.322e-11, 5.521e-11, 5.725e-11, 5.827e-11, 5.685e-11,
02622
02623
            5.245e-11, 4.612e-11, 3.884e-11, 3.129e-11, 2.404e-11, 1.732e-11,
             1.223e-11, 8.574e-12, 5.888e-12, 3.986e-12, 2.732e-12, 1.948e-12,
             1.414e-12, 1.061e-12, 8.298e-13, 6.612e-13, 5.413e-13, 4.472e-13,
02625
02626
            3.772e-13, 3.181e-13, 2.645e-13, 2.171e-13, 1.778e-13, 1.464e-13,
02627
            1.183e-13, 9.637e-14, 7.991e-14, 6.668e-14, 5.57e-14, 4.663e-14,
            3.848e-14, 3.233e-14, 2.706e-14, 2.284e-14, 1.944e-14, 1.664e-14, 1.43e-14, 1.233e-14, 1.066e-14, 9.234e-15, 8.023e-15, 6.993e-15, 6.119e-15, 5.384e-15, 4.774e-15, 4.283e-15, 3.916e-15, 3.695e-15,
02628
02629
02630
            3.682e-15, 4.004e-15, 4.912e-15, 6.853e-15, 1.056e-14, 1.712e-14,
02631
02632
             2.804e-14, 4.516e-14, 7.113e-14, 1.084e-13, 1.426e-13, 1.734e-13,
02633
            1.978e-13, 2.194e-13, 2.388e-13, 2.489e-13, 2.626e-13, 2.865e-13,
             3.105e-13, 3.387e-13, 3.652e-13, 3.984e-13, 4.398e-13, 4.906e-13,
02634
            5.55e-13, 6.517e-13, 7.813e-13, 9.272e-13, 1.164e-12, 1.434e-12, 1.849e-12, 2.524e-12, 3.328e-12, 4.523e-12, 6.108e-12, 8.207e-12,
02635
             1.122e-11, 1.477e-11, 1.9e-11, 2.412e-11, 2.984e-11, 3.68e-11,
02637
02638
             4.353e-11, 4.963e-11, 5.478e-11, 5.903e-11, 6.233e-11, 6.483e-11,
            6.904e-11, 7.569e-11, 8.719e-11, 1.048e-10, 1.278e-10, 1.557e-10, 1.869e-10, 2.218e-10, 2.61e-10, 2.975e-10, 3.371e-10, 3.746e-10, 4.065e-10, 4.336e-10, 4.503e-10, 4.701e-10, 4.8e-10, 4.917e-10,
02639
02640
02641
```

```
5.038e-10, 5.128e-10, 5.143e-10, 5.071e-10, 5.019e-10, 5.025e-10,
           5.183e-10, 5.496e-10, 5.877e-10, 6.235e-10, 6.42e-10, 6.234e-10, 5.698e-10, 4.916e-10, 4.022e-10, 3.126e-10, 2.282e-10, 1.639e-10,
02643
02644
02645
           1.142 e^{-10},\ 7.919 e^{-11},\ 5.69 e^{-11},\ 4.313 e^{-11},\ 3.413 e^{-11},\ 2.807 e^{-11},
           2.41e-11, 2.166e-11, 2.024e-11, 1.946e-11, 1.929e-11, 1.963e-11, 2.035e-11, 2.162e-11, 2.305e-11, 2.493e-11, 2.748e-11, 3.048e-11,
02646
02647
           3.413e-11, 3.754e-11, 4.155e-11, 4.635e-11, 5.11e-11, 5.734e-11, 6.338e-11, 6.99e-11, 7.611e-11, 8.125e-11, 8.654e-11, 8.951e-11,
02649
02650
           9.182e-11, 9.31e-11, 9.273e-11, 9.094e-11, 8.849e-11, 8.662e-11,
02651
           8.67e-11, 8.972e-11, 9.566e-11, 1.025e-10, 1.083e-10, 1.111e-10,
           1.074e-10, 9.771e-11, 8.468e-11, 6.958e-11, 5.47e-11, 4.04e-11,
02652
           2.94e-11, 2.075e-11, 1.442e-11, 1.01e-11, 7.281e-12, 5.409e-12,
02653
           4.138e-12, 3.304e-12, 2.784e-12, 2.473e-12, 2.273e-12, 2.186e-12,
02654
           2.118e-12, 2.066e-12, 1.958e-12, 1.818e-12, 1.675e-12, 1.509e-12,
02655
02656
           1.349e-12, 1.171e-12, 9.838e-13, 8.213e-13, 6.765e-13, 5.378e-13,
02657
           4.161e-13, 3.119e-13, 2.279e-13, 1.637e-13, 1.152e-13, 8.112e-14,
           5.919e-14, 4.47e-14, 3.492e-14, 2.811e-14, 2.319e-14, 1.948e-14, 1.66e-14, 1.432e-14, 1.251e-14, 1.109e-14, 1.006e-14, 9.45e-15,
02658
02659
           9.384e-15, 1.012e-14, 1.216e-14, 1.636e-14, 2.305e-14, 3.488e-14,
02660
           5.572e-14, 8.479e-14, 1.265e-13, 1.905e-13, 2.73e-13, 3.809e-13,
           4.955e-13, 6.303e-13, 7.861e-13, 9.427e-13, 1.097e-12, 1.212e-12,
02662
02663
           1.328e-12, 1.415e-12, 1.463e-12, 1.495e-12, 1.571e-12, 1.731e-12,
           1.981e-12, 2.387e-12, 2.93e-12, 3.642e-12, 4.584e-12, 5.822e-12,
02664
           7.278e-12, 9.193e-12, 1.135e-11, 1.382e-11, 1.662e-11, 1.958e-11, 2.286e-11, 2.559e-11, 2.805e-11, 2.988e-11, 3.106e-11, 3.182e-11,
02665
02666
           3.2e-11, 3.258e-11, 3.362e-11, 3.558e-11, 3.688e-11, 3.8e-11,
           3.929e-11, 4.062e-11, 4.186e-11, 4.293e-11, 4.48e-11, 4.643e-11,
02668
02669
           4.704e-11, 4.571e-11, 4.206e-11, 3.715e-11, 3.131e-11, 2.541e-11,
02670
           1.978e-11, 1.508e-11, 1.146e-11, 8.7e-12, 6.603e-12, 5.162e-12,
           4.157e-12, 3.408e-12, 2.829e-12, 2.405e-12, 2.071e-12, 1.826e-12,
02671
02672
           1.648e-12, 1.542e-12, 1.489e-12, 1.485e-12, 1.493e-12, 1.545e-12,
           1.637e-12, 1.814e-12, 2.061e-12, 2.312e-12, 2.651e-12, 3.03e-12,
           3.46e-12, 3.901e-12, 4.306e-12, 4.721e-12, 5.008e-12, 5.281e-12,
02674
02675
           5.541e-12, 5.791e-12, 6.115e-12, 6.442e-12, 6.68e-12, 6.791e-12,
           6.831e-12, 6.839e-12, 6.946e-12, 7.128e-12, 7.537e-12, 8.036e-12, 8.392e-12, 8.526e-12, 8.11e-12, 7.325e-12, 6.329e-12, 5.183e-12,
02676
02677
02678
           4.081e-12, 2.985e-12, 2.141e-12, 1.492e-12, 1.015e-12, 6.684e-13,
           4.414e-13, 2.987e-13, 2.038e-13, 1.391e-13, 9.86e-14, 7.24e-14,
02680
           5.493e-14, 4.288e-14, 3.427e-14, 2.787e-14, 2.296e-14, 1.909e-14,
           1.598e-14, 1.344e-14, 1.135e-14, 9.616e-15, 8.169e-15, 6.957e-15,
02681
02682
           5.938e-15, 5.08e-15, 4.353e-15, 3.738e-15, 3.217e-15, 2.773e-15,
           2.397e-15, 2.077e-15, 1.805e-15, 1.575e-15, 1.382e-15, 1.221e-15,
02683
           1.09e-15, 9.855e-16, 9.068e-16, 8.537e-16, 8.27e-16, 8.29e-16, 8.634e-16, 9.359e-16, 1.055e-15, 1.233e-15, 1.486e-15, 1.839e-15,
02684
02685
           2.326e-15, 2.998e-15, 3.934e-15, 5.256e-15, 7.164e-15, 9.984e-15,
02687
           1.427e-14, 2.099e-14, 3.196e-14, 5.121e-14, 7.908e-14, 1.131e-13,
02688
           1.602e-13, 2.239e-13, 3.075e-13, 4.134e-13, 5.749e-13, 7.886e-13,
02689
           1.071e-12,\ 1.464e-12,\ 2.032e-12,\ 2.8e-12,\ 3.732e-12,\ 4.996e-12,
           6.483e-12, 8.143e-12, 1.006e-11, 1.238e-11, 1.484e-11, 1.744e-11,
02690
           2.02e-11, 2.274e-11, 2.562e-11, 2.848e-11, 3.191e-11, 3.617e-11,
02691
           4.081e-11, 4.577e-11, 4.937e-11, 5.204e-11, 5.401e-11, 5.462e-11,
           5.507e-11, 5.51e-11, 5.605e-11, 5.686e-11, 5.739e-11, 5.766e-11,
02693
02694
           5.74e-11, 5.754e-11, 5.761e-11, 5.777e-11, 5.712e-11, 5.51e-11,
           5.088e-11, 4.438e-11, 3.728e-11, 2.994e-11, 2.305e-11, 1.715e-11, 1.256e-11, 9.208e-12, 6.745e-12, 5.014e-12, 3.785e-12, 2.9e-12,
02695
02696
02697
           2.239e-12, 1.757e-12, 1.414e-12, 1.142e-12, 9.482e-13, 8.01e-13,
           6.961e-13, 6.253e-13, 5.735e-13, 5.433e-13, 5.352e-13, 5.493e-13,
           5.706e-13, 6.068e-13, 6.531e-13, 7.109e-13, 7.767e-13, 8.59e-13,
02699
02700
           9.792e-13, 1.142e-12, 1.371e-12, 1.65e-12, 1.957e-12, 2.302e-12,
02701
           2.705e-12, 3.145e-12, 3.608e-12, 4.071e-12, 4.602e-12, 5.133e-12,
02702
           5.572e-12, 5.987e-12, 6.248e-12, 6.533e-12, 6.757e-12, 6.935e-12,
02703
           7.224e-12, 7.422e-12, 7.538e-12, 7.547e-12, 7.495e-12, 7.543e-12,
02704
           7.725e-12, 8.139e-12, 8.627e-12, 9.146e-12, 9.443e-12, 9.318e-12,
           8.649e-12, 7.512e-12, 6.261e-12, 4.915e-12, 3.647e-12, 2.597e-12,
02705
02706
           1.785e-12, 1.242e-12, 8.66e-13, 6.207e-13, 4.61e-13, 3.444e-13,
02707
           2.634e-13, 2.1e-13, 1.725e-13, 1.455e-13, 1.237e-13, 1.085e-13,
           9.513e-14, 7.978e-14, 6.603e-14, 5.288e-14, 4.084e-14, 2.952e-14,
02708
02709
           2.157e-14, 1.593e-14, 1.199e-14, 9.267e-15, 7.365e-15, 6.004e-15,
02710
           4.995e-15, 4.218e-15, 3.601e-15, 3.101e-15, 2.692e-15, 2.36e-15,
           2.094e-15, 1.891e-15, 1.755e-15, 1.699e-15, 1.755e-15, 1.987e-15,
02712
           2.506e-15, 3.506e-15, 5.289e-15, 8.311e-15, 1.325e-14, 2.129e-14,
02713
           3.237e-14, 4.595e-14, 6.441e-14, 8.433e-14, 1.074e-13, 1.383e-13,
           1.762e-13, 2.281e-13, 2.831e-13, 3.523e-13, 4.38e-13, 5.304e-13, 6.29e-13, 7.142e-13, 8.032e-13, 8.934e-13, 9.888e-13, 1.109e-12, 1.261e-12, 1.462e-12, 1.74e-12, 2.099e-12, 2.535e-12, 3.008e-12,
02714
02715
02716
           3.462e-12, 3.856e-12, 4.098e-12, 4.239e-12, 4.234e-12, 4.132e-12,
02717
           3.986e-12, 3.866e-12, 3.829e-12, 3.742e-12, 3.705e-12, 3.694e-12,
02718
02719
           3.765e-12, 3.849e-12, 3.929e-12, 4.056e-12, 4.092e-12, 4.047e-12,
02720
           3.792e-12, 3.407e-12, 2.953e-12, 2.429e-12, 1.931e-12, 1.46e-12,
           1.099e-12, 8.199e-13, 6.077e-13, 4.449e-13, 3.359e-13, 2.524e-13,
02721
           1.881e-13, 1.391e-13, 1.02e-13, 7.544e-14, 5.555e-14, 4.22e-14,
02722
           3.321e-14, 2.686e-14, 2.212e-14, 1.78e-14, 1.369e-14, 1.094e-14,
           9.13e-15, 8.101e-15, 7.828e-15, 8.393e-15, 1.012e-14, 1.259e-14,
02724
02725
           1.538e-14, 1.961e-14, 2.619e-14, 3.679e-14, 5.049e-14, 6.917e-14,
02726
           8.88e-14,\ 1.115e-13,\ 1.373e-13,\ 1.619e-13,\ 1.878e-13,\ 2.111e-13,
           2.33e-13, 2.503e-13, 2.613e-13, 2.743e-13, 2.826e-13, 2.976e-13, 3.162e-13, 3.36e-13, 3.491e-13, 3.541e-13, 3.595e-13, 3.608e-13,
02727
02728
```

```
3.709e-13, 3.869e-13, 4.12e-13, 4.366e-13, 4.504e-13, 4.379e-13,
            3.955e-13, 3.385e-13, 2.741e-13, 2.089e-13, 1.427e-13, 9.294e-14, 5.775e-14, 3.565e-14, 2.21e-14, 1.398e-14, 9.194e-15, 6.363e-15,
02730
02731
02732
            4.644e-15, 3.55e-15, 2.808e-15, 2.274e-15, 1.871e-15, 1.557e-15,
02733
            1.308e-15, 1.108e-15, 9.488e-16, 8.222e-16, 7.238e-16, 6.506e-16, 6.008e-16, 5.742e-16, 5.724e-16, 5.991e-16, 6.625e-16, 7.775e-16,
02734
            9.734e-16, 1.306e-15, 1.88e-15, 2.879e-15, 4.616e-15, 7.579e-15,
            1.248e-14, 2.03e-14, 3.244e-14, 5.171e-14, 7.394e-14, 9.676e-14,
02736
            1.199e-13, 1.467e-13, 1.737e-13, 2.02e-13, 2.425e-13, 3.016e-13, 3.7e-13, 4.617e-13, 5.949e-13, 7.473e-13, 9.378e-13, 1.191e-12, 1.481e-12, 1.813e-12, 2.232e-12, 2.722e-12, 3.254e-12, 3.845e-12, 4.458e-12, 5.048e-12, 5.511e-12, 5.898e-12, 6.204e-12, 6.293e-12,
02737
02738
02739
02740
            6.386e-12, 6.467e-12, 6.507e-12, 6.466e-12, 6.443e-12, 6.598e-12, 6.873e-12, 7.3e-12, 7.816e-12, 8.368e-12, 8.643e-12, 8.466e-12,
02741
02742
02743
            7.871e-12, 6.853e-12, 5.714e-12, 4.482e-12, 3.392e-12, 2.613e-12,
02744
            2.008e-12, 1.562e-12, 1.228e-12, 9.888e-13, 7.646e-13, 5.769e-13,
02745
            4.368e-13, 3.324e-13, 2.508e-13, 1.916e-13
02746
         };
02748
         static double xfcrev[15] =
02749
            { 1.003, 1.009, 1.015, 1.023, 1.029, 1.033, 1.037,
02750
            1.039, 1.04, 1.046, 1.036, 1.027, 1.01, 1.002, 1.
02751
02752
02753
         double a1, a2, a3, dw, ew, dx, xw, xx, vf2, vf6, cw260, cw296,
02754
           sfac, fscal, cwfrn, ctmpth, ctwfrn, ctwslf;
02755
02756
         int iw, jw, ix;
02757
02758
         /* Get H2O continuum absorption... */
02759
         xw = nu / 10 + 1;
02760
         if (xw >= 1 && xw < 2001) {
02761
           iw = (int) xw;
02762
            jw = iw + 1;
            dw = xw - iw;

ew = 1 - dw;
02763
02764
            cw296 = ew * h2o296[iw - 1] + dw * h2o296[jw - 1];
02765
            cw260 = ew * h2o260[iw - 1] + dw * h2o260[jw - 1];
02766
            cwfrn = ew * h2ofrn[iw - 1] + dw * h2ofrn[jw - 1];
02767
02768
            if (nu <= 820 || nu >= 960) {
02769
              sfac = 1;
02770
            } else {
02771
             xx = (nu - 820) / 10:
02772
              ix = (int) xx;
              dx = xx - ix;
02773
02774
              sfac = (1 - dx) * xfcrev[ix] + dx * xfcrev[ix + 1];
02775
            ctwslf = sfac * cw296 * pow(cw260 / cw296, (296 - t) / (296 - 260));
02776
            vf2 = gsl_pow_2(nu - 370);
02777
            vf6 = gsl_pow_3(vf2);
02778
            fscal = 36100 / (vf2 + vf6 * 1e-8 + 36100) * -.25 + 1;
02780
            ctwfrn = cwfrn * fscal;
            a1 = nu * u * tanh(.7193876 / t * nu);
a2 = 296 / t;
a3 = p / P0 * (q * ctwslf + (1 - q) * ctwfrn) * 1e-20;
02781
02782
02783
02784
            ctmpth = a1 * a2 * a3;
02785
         } else
02786
            ctmpth = 0;
02787
         return ctmpth;
02788 3
```

## 5.15.2.8 double ctmn2 ( double nu, double p, double t )

Compute nitrogen continuum (absorption coefficient).

Definition at line 2792 of file jurassic.c.

```
02795
02796
       static double ba[98] = { 0., 4.45e-8, 5.22e-8, 6.46e-8, 7.75e-8, 9.03e-8,
02797
         1.06e-7, 1.21e-7, 1.37e-7, 1.57e-7, 1.75e-7, 2.01e-7, 2.3e-7,
02798
          2.59e-7, 2.95e-7, 3.26e-7, 3.66e-7, 4.05e-7, 4.47e-7, 4.92e-7,
02799
         5.34e-7, 5.84e-7, 6.24e-7, 6.67e-7, 7.14e-7, 7.26e-7, 7.54e-7,
         7.84e-7, 8.09e-7, 8.42e-7, 8.62e-7, 8.87e-7, 9.11e-7, 9.36e-7,
02801
02802
         9.76e-7, 1.03e-6, 1.11e-6, 1.23e-6, 1.39e-6, 1.61e-6, 1.76e-6,
02803
         1.94e-6, 1.97e-6, 1.87e-6, 1.75e-6, 1.56e-6, 1.42e-6, 1.35e-6,
02804
         1.32e-6, 1.29e-6, 1.29e-6, 1.3e-6, 1.32e-6, 1.33e-6,
02805
         1.34e-6, 1.35e-6, 1.33e-6, 1.31e-6, 1.29e-6, 1.24e-6, 1.2e-6,
02806
         1.16e-6, 1.1e-6, 1.04e-6, 9.96e-7, 9.38e-7, 8.63e-7, 7.98e-7,
         7.26e-7, 6.55e-7, 5.94e-7, 5.35e-7, 4.74e-7, 4.24e-7, 3.77e-7,
```

```
3.33e-7, 2.96e-7, 2.63e-7, 2.34e-7, 2.08e-7, 1.85e-7, 1.67e-7,
                1.47e-7, 1.32e-7, 1.2e-7, 1.09e-7, 9.85e-8, 9.08e-8, 8.18e-8,
02809
02810
                7.56e-8, 6.85e-8, 6.14e-8, 5.83e-8, 5.77e-8, 5e-8, 4.32e-8, 0.
02811
02812
             static double betaa[98] = { 802., 802., 761., 722., 679., 646., 609., 562.,
511., 472., 436., 406., 377., 355., 338., 319., 299., 278., 255.,
233., 208., 184., 149., 107., 66., 25., -13., -49., -82., -104.,
02813
02815
               233., 208., 184., 149., 107., 66., 25., -13., -49., -82., -104., -119., -130., -139., -144., -146., -146., -147., -148., -150., -153., -160., -169., -181., -189., -195., -200., -205., -209., -211., -210., -210., -209., -205., -199., -190., -180., -168., -157., -143., -126., -108., -89., -63., -32., 1., 35., 65., 95., 121., 141., 152., 161., 164., 164., 161., 155., 148., 143., 137., 133., 131., 133., 139., 150., 165., 187., 213., 248., 284., 321., 372., 449., 514., 569., 609., 642., 673., 673.
02816
02817
02818
02819
02820
02821
02822
02823
02824
            static double nua[98] = { 2120., 2125., 2130., 2135., 2140., 2145., 2150.,
2155., 2160., 2165., 2170., 2175., 2180., 2185., 2190., 2195.,
2200., 2205., 2210., 2215., 2220., 2225., 2230., 2235., 2240.,
02825
02826
                2245., 2250., 2255., 2260., 2265., 2270., 2275., 2280., 2285., 2290., 2295., 2300., 2305., 2310., 2315., 2320., 2325., 2330.,
02828
02829
02830
                2335., 2340., 2345., 2350., 2355., 2360., 2365., 2370., 2375.,
                2380., 2385., 2390., 2395., 2400., 2405., 2410., 2415., 2420., 2425., 2430., 2435., 2440., 2445., 2450., 2455., 2460., 2465.,
02831
02832
                2470., 2475., 2480., 2485., 2490., 2495., 2500., 2505., 2510., 2515., 2520., 2525., 2530., 2535., 2540., 2545., 2550., 2555.,
02834
02835
                2560., 2565., 2570., 2575., 2580., 2585., 2590., 2595., 2600., 2605.
02836
02837
02838
            double b, beta, q_n2 = 0.79, t0 = 273, tr = 296;
02839
02840
             int idx;
02841
02842
             /* Check wavenumber range...
             if (nu < nua[0] || nu > nua[97])
02843
02844
              return 0;
02846
             /* Interpolate B and beta... */
02847
            idx = locate(nua, 98, nu);
02848
            b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
            beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02849
02850
02851
             /* Compute absorption coefficient... */
            return 0.1 * gsl_pow_2(p / P0) * gsl_pow_2(t0 / t) 
* exp(beta * (1 / tr - 1 / t))
02853
02854
                * q_n2 * b * (q_n2 + (1 - q_n2) * (1.294 - 0.4545 * t / tr));
02855 }
```

Here is the call graph for this function:



## 5.15.2.9 double ctmo2 ( double nu, double p, double t )

Compute oxygen continuum (absorption coefficient).

Definition at line 2859 of file jurassic.c.

```
02862 {
02863
02864 static double ba[90] = { 0., .061, .074, .084, .096, .12, .162, .208, .246,
02865 .285, .314, .38, .444, .5, .571, .673, .768, .853, .966, 1.097,
02866 1.214, 1.333, 1.466, 1.591, 1.693, 1.796, 1.922, 2.037, 2.154,
```

```
2.264, 2.375, 2.508, 2.671, 2.847, 3.066, 3.417, 3.828, 4.204,
              4.453, 4.599, 4.528, 4.284, 3.955, 3.678, 3.477, 3.346, 3.29, 3.251, 3.231, 3.226, 3.212, 3.192, 3.108, 3.033, 2.911, 2.798,
02868
02869
              2.646, 2.508, 2.322, 2.13, 1.928, 1.757, 1.588, 1.417, 1.253, 1.109, .99, .888, .791, .678, .587, .524, .464, .403, .357, .32.29, .267, .242, .215, .182, .16, .146, .128, .103, .087, .081, .071, .064, 0.
02870
02871
                                                                                     .403, .357, .32,
02872
02874
02875
           static double betaa[90] = { 467., 467., 400., 315., 379., 368., 475., 521., 531., 512., 442., 444., 430., 381., 335., 324., 296., 248., 215., 193., 158., 127., 101., 71., 31., -6., -26., -47., -63., -79., -88., -88., -87., -90., -98., -99., -109., -134., -160., -167., -164., -158., -153., -151., -156., -166., -168., -173., -170., -161., -145., -126., -108., -84., -59., -29., 4., 41., 73., 97., 123.
02876
02877
02878
02879
02880
02881
              123., 159., 198., 220., 242., 256., 281., 311., 334., 319., 313., 321., 323., 310., 315., 320., 335., 361., 378., 373., 338., 319.,
02882
02883
02884
              346., 322., 291., 290., 350., 371., 504., 504.
02885
02886
02887
           static double nua[90] = { 1360., 1365., 1370., 1375., 1380., 1385., 1390.,
02888
              1395., 1400., 1405., 1410., 1415., 1420., 1425., 1430., 1435.,
02889
               1440., 1445., 1450., 1455., 1460., 1465., 1470., 1475., 1480.,
              1485., 1490., 1495., 1500., 1505., 1510., 1515., 1520., 1525., 1530., 1535., 1540., 1545., 1550., 1555., 1560., 1565., 1570.,
02890
02891
               1575., 1580., 1585., 1590., 1595., 1600., 1605., 1610., 1615.,
02893
               1620., 1625., 1630., 1635., 1640., 1645., 1650., 1655., 1660.,
02894
              1665., 1670., 1675., 1680., 1685., 1690., 1695., 1700., 1705.,
02895
              1710., 1715., 1720., 1725., 1730., 1735., 1740., 1745., 1750.,
               1755., 1760., 1765., 1770., 1775., 1780., 1785., 1790., 1795.,
02896
02897
              1800., 1805.
02898
02899
02900
           double b, beta, q_02 = 0.21, t0 = 273, tr = 296;
02901
02902
           int idx:
02903
            /* Check wavenumber range...
02905
           if (nu < nua[0] || nu > nua[89])
02906
             return 0;
02907
02908
           /* Interpolate B and beta... */
           idx = locate(nua, 90, nu);
b = LIN(nua[idx], ba[idx], nua[idx + 1], ba[idx + 1], nu);
02909
02910
02911
           beta = LIN(nua[idx], betaa[idx], nua[idx + 1], betaa[idx + 1], nu);
02912
02913
            /* Compute absorption coefficient... */
           return 0.1 * gsl_pow_2(p / P0) * gsl_pow_2(t0 / t)
 * exp(beta * (1 / tr - 1 / t)) * q_o2 * b;
02914
02915
02916 }
```

Here is the call graph for this function:



5.15.2.10 void copy\_atm ( ctl\_t \* ctl, atm\_t \* atm\_dest, atm\_t \* atm\_src, int init )

Copy and initialize atmospheric data.

Definition at line 2920 of file jurassic.c.

```
02924 {
02925
02926 int ig, ip, iw;
```

```
02927
02928
        size t s;
02929
02930
        /* Data size... */
02931
        s = (size_t) atm_src->np * sizeof(double);
02932
        /* Copy data... */
02934
        atm_dest->np = atm_src->np;
02935
        memcpy(atm_dest->time, atm_src->time, s);
02936
        memcpy(atm_dest->z, atm_src->z, s);
02937
        memcpy(atm_dest->lon, atm_src->lon, s);
02938
        memcpy(atm_dest->lat, atm_src->lat, s);
        memcpy(atm_dest->p, atm_src->p, s);
02939
02940
        memcpy(atm_dest->t, atm_src->t, s);
02941
        for (ig = 0; ig < ctl->ng; ig++)
02942
          memcpy(atm_dest->q[ig], atm_src->q[ig], s);
        for (iw = 0; iw < ctl->nw; iw++)
02943
02944
         memcpy(atm_dest->k[iw], atm_src->k[iw], s);
02945
02946
        /* Initialize... */
        if (init)
02947
        for (ip = 0; ip < atm_dest->np; ip++) {
02948
           atm_dest->p[ip] = 0;
atm_dest->t[ip] = 0;
02949
02950
02951
           for (ig = 0; ig < ctl->ng; ig++)
02952
            atm_dest->q[ig][ip] = 0;
for (iw = 0; iw < ctl->nw; iw++)
02953
             atm_dest->k[iw][ip] = 0;
02954
          }
02955
02956 }
```

```
5.15.2.11 void copy_obs ( ctl t * ctl, obs t * obs_dest, obs t * obs_src, int init )
```

Copy and initialize observation data.

Definition at line 2960 of file jurassic.c.

```
02964
                     {
02965
        int id, ir;
02967
02968
        size_t s;
02969
02970
        /* Data size... */
02971
        s = (size_t) obs_src->nr * sizeof(double);
02972
02973
        /* Copy data... */
02974
        obs_dest->nr = obs_src->nr;
02975
         memcpy(obs_dest->time, obs_src->time, s);
        memcpy(obs_dest->obsz, obs_src->obsz, s);
memcpy(obs_dest->obslon, obs_src->obslon, s);
memcpy(obs_dest->obslat, obs_src->obslat, s);
02976
02977
02978
02979
        memcpy(obs_dest->vpz, obs_src->vpz, s);
02980
         memcpy(obs_dest->vplon, obs_src->vplon, s);
         memcpy(obs_dest->vplat, obs_src->vplat, s);
02981
02982
        memcpy(obs_dest->tpz, obs_src->tpz, s);
        memcpy(obs_dest->tplon, obs_src->tplon, s);
memcpy(obs_dest->tplat, obs_src->tplat, s);
02983
02984
02985
        for (id = 0; id < ctl->nd; id++)
02986
          memcpy(obs_dest->rad[id], obs_src->rad[id], s);
02987
        for (id = 0; id < ctl->nd; id++)
          memcpy(obs_dest->tau[id], obs_src->tau[id], s);
02988
02989
        /* Initialize... */
02991
        if (init)
02992
         for (id = 0; id < ctl->nd; id++)
02993
              for (ir = 0; ir < obs_dest->nr; ir++)
02994
               if (gsl_finite(obs_dest->rad[id][ir])) {
                 obs_dest->rad[id][ir] = 0;
obs_dest->tau[id][ir] = 0;
02995
02996
02997
02998 }
```

```
5.15.2.12 int find_emitter ( ctl_t * ctl, const char * emitter )
```

Find index of an emitter.

Definition at line 3002 of file jurassic.c.

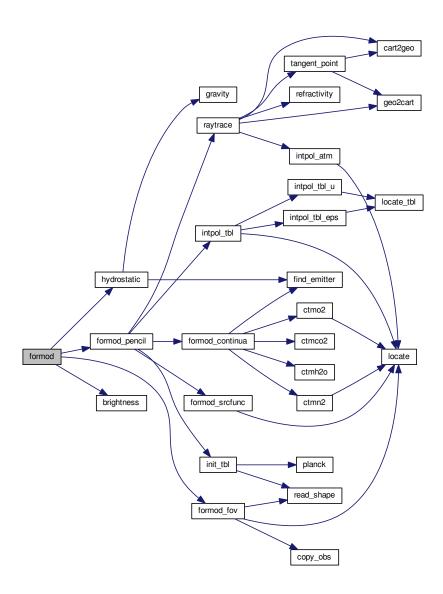
```
5.15.2.13 void formod ( ctl_t * ctl, atm_t * atm, obs_t * obs )
```

Determine ray paths and compute radiative transfer.

Definition at line 3017 of file jurassic.c.

```
03020
03021
03022
        int id, ir, *mask;
03023
03024
         /* Allocate... */
        ALLOC(mask, int,
ND * NR);
03025
03026
03027
        /\star Save observation mask... \star/
03028
        for (id = 0; id < ctl->nd; id+)
  for (ir = 0; ir < obs->nr; ir++)
    mask[id * NR + ir] = !gsl_finite(obs->rad[id][ir]);
03029
03030
03031
03032
03033
         /* Hydrostatic equilibrium... */
03034
        hydrostatic(ctl, atm);
03035
03036
         /* Claculate pencil beams... */
        for (ir = 0; ir < obs->nr; ir++)
03037
03038
           formod_pencil(ctl, atm, obs, ir);
03039
03040
        /* Apply field-of-view convolution... */
03041
        formod_fov(ctl, obs);
03042
03043
         /* Convert radiance to brightness temperature... */
03044
         if (ctl->write_bbt)
03045
         for (id = 0; id < ctl->nd; id++)
03046
             for (ir = 0; ir < obs->nr; ir++)
                obs->rad[id][ir] = brightness(obs->rad[id][ir], ctl->nu[id]);
03047
03048
        /* Apply observation mask... */
for (id = 0; id < ctl->nd; id++)
  for (ir = 0; ir < obs->nr; ir++)
03049
03050
03051
03052
             if (mask[id * NR + ir])
                obs->rad[id][ir] = GSL_NAN;
03053
03054
03055
         /* Free... */
03056
        free(mask);
03057 }
```

Here is the call graph for this function:



5.15.2.14 void formod\_continua (  $ctl_t * ctl$ ,  $los_t * los$ , int ip, double \* beta )

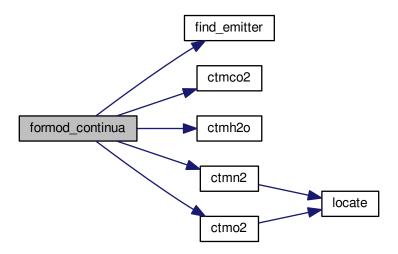
Compute absorption coefficient of continua.

Definition at line 3061 of file jurassic.c.

```
03065 {
03066
03067 static int ig_co2 = -999, ig_h2o = -999;
03068
03069 int id;
03070
03071 /* Extinction... */
03072 for (id = 0; id < ctl->nd; id++)
03073 beta[id] = los->k[ctl->window[id]][ip];
03074
03075 /* CO2 continuum... */
03076 if (ctl->ctm_co2) {
03077 if (ig_co2 == -999)
```

```
ig_co2 = find_emitter(ctl, "CO2");
           if (ig_co2 >= 0)
for (id = 0; id < ctl->nd; id++)
03079
03080
               beta[id] += ctmco2(ctl->nu[id], los->p[ip], los->t[ip],
03081
                                      los->u[ig_co2][ip]) / los->ds[ip];
03082
03083
03084
03085
         /* H2O continuum... */
03086
         if (ctl->ctm_h2o) {
          if (ig_h2o == -999)
  ig_h2o = find_emitter(ctl, "H2O");
03087
03088
           if (ig_h2o >= 0)
  for (id = 0; id < ctl->nd; id++)
  beta[id] += ctmh2o(ctl->nu[id], los->p[ip], los->t[ip],
03089
03090
03091
03092
                                      los->q[ig_h2o][ip],
                                       los->u[ig_h2o][ip]) / los->ds[ip];
03093
03094
03095
03096
         /* N2 continuum... */
03097
         if (ctl->ctm_n2)
          for (id = 0; id < ct1->nd; id++)
  beta[id] += ctmn2(ct1->nu[id], los->p[ip], los->t[ip]);
03098
03099
03100
         /* 02 continuum... */
0.3101
03102
         if (ctl->ctm_o2)
03103
           for (id = 0; id < ctl->nd; id++)
03104
              beta[id] \textit{ += } ctmo2(ctl->nu[id], los->p[ip], los->t[ip]);
03105 }
```

Here is the call graph for this function:



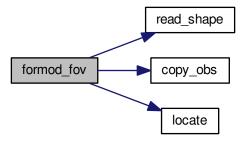
5.15.2.15 void formod\_fov (  $ctl\_t * ctl$ , obs\_t \* obs )

Apply field of view convolution.

Definition at line 3109 of file jurassic.c.

```
03118
03119
        double rad[ND][NR], tau[ND][NR], wsum, z[NR], zfov;
03120
03121
        int i, id, idx, ir, ir2, nz;
03122
03123
        /* Do not take into account FOV... */
03124
        if (ctl->fov[0] == '-')
03125
03126
        /* Initialize FOV data... */
if (!init) {
03127
03128
03129
         init = 1;
03130
          read_shape(ctl->fov, dz, w, &n);
03131
03132
03133
        /* Allocate... */
03134
        ALLOC(obs2, obs_t, 1);
03135
03136
        /* Copy observation data... */
03137
        copy_obs(ctl, obs2, obs, 0);
03138
03139
        /* Loop over ray paths... */
        for (ir = 0; ir < obs->nr; ir++) {
0.3140
0.3141
03142
           /* Get radiance and transmittance profiles... */
03143
           for (ir2 = GSL_MAX(ir - NFOV, 0); ir2 < GSL_MIN(ir + 1 + NFOV, obs->nr);
03144
             ir2++)
if (obs->time[ir2] == obs->time[ir]) {
03145
03146
03147
              z[nz] = obs2->vpz[ir2];
for (id = 0; id < ct1->nd; id++) {
03148
                rad[id][nz] = obs2->rad[id][ir2];
tau[id][nz] = obs2->tau[id][ir2];
03149
03150
03151
03152
              nz++;
03153
          if (nz < 2)
03154
            ERRMSG("Cannot apply FOV convolution!");
03155
03156
03157
           /\star Convolute profiles with FOV... \star/
          wsum = 0;
for (id = 0; id < ctl->nd; id++) {
03158
03159
            obs->rad[id][ir] = 0;
0.3160
03161
            obs->tau[id][ir] = 0;
03162
03163
           for (i = 0; i < n; i++) {</pre>
           zfov = obs->vpz[ir] + dz[i];
idx = locate(z, nz, zfov);
03164
03165
             for (id = 0; id < ctl->nd; id++) {
03166
              obs->rad[id][ir] += w[i]
03167
               * LIN(z[idx], rad[id][idx], z[idx + 1], rad[id][idx + 1], zfov);
obs->tau[id][ir] += w[i]
03168
03169
03170
                 * LIN(z[idx], tau[id][idx], z[idx + 1], tau[id][idx + 1], zfov);
0.3171
03172
             wsum += w[i];
03173
03174
          for (id = 0; id < ctl->nd; id++) {
03175
             obs->rad[id][ir] /= wsum;
03176
             obs->tau[id][ir] /= wsum;
03177
03178
        }
03179
03180
         /* Free... */
        free(obs2);
03182 }
```

Here is the call graph for this function:



5.15.2.16 void formod\_pencil (  $ctl_t * ctl$ ,  $atm_t * atm$ ,  $obs_t * obs$ , int ir )

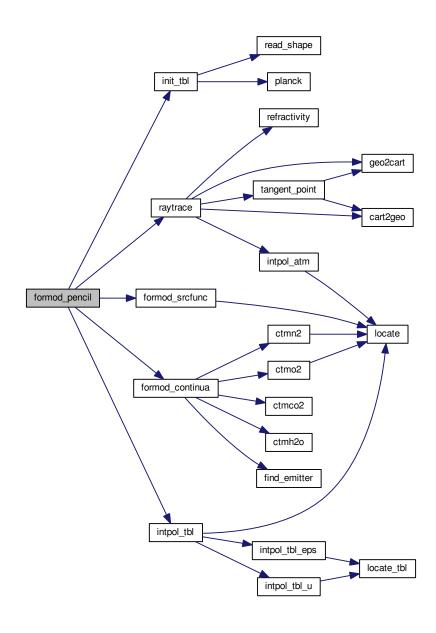
Compute radiative transfer for a pencil beam.

Definition at line 3186 of file jurassic.c.

```
03190
03191
03192
        static tbl_t *tbl;
03193
03194
        static int init = 0;
03195
03196
        los t *los:
03197
03198
        double beta_ctm[ND], eps, src_planck[ND], tau_path[NG][ND], tau_gas[ND];
03199
03200
         int id, ip;
03201
03202
         /* Initialize look-up tables... */
03203
         if (!init) {
03204
          init = 1;
03205
           ALLOC(tbl, tbl_t, 1);
03206
           init_tbl(ctl, tbl);
03207
03208
        /* Allocate... */
ALLOC(los, los_t, 1);
03209
03210
03211
         /* Initialize... */
for (id = 0; id < ctl->nd; id++) {
  obs->rad[id][ir] = 0;
03212
03213
03214
03215
          obs->tau[id][ir] = 1;
03216
03217
03218
         /* Raytracing... */
03219
         raytrace(ctl, atm, obs, los, ir);
03220
         /* Loop over LOS points... */
03221
03222
         for (ip = 0; ip < los->np; ip++) {
03223
03224
           /* Get trace gas transmittance... */
03225
           intpol_tbl(ctl, tbl, los, ip, tau_path, tau_gas);
03226
03227
           /* Get continuum absorption... */
03228
           formod_continua(ctl, los, ip, beta_ctm);
03229
03230
           /* Compute Planck function... */
03231
           formod_srcfunc(ctl, tbl, los->t[ip], src_planck);
03232
03233
           /* Loop over channels... */
for (id = 0; id < ctl->nd; id++)
    if (tau_gas[id] > 0) {
03234
03235
03236
```

```
/* Get segment emissivity... */
03238
            eps = 1 - tau_gas[id] * exp(-beta_ctm[id] * los->ds[ip]);
03239
03240
            /* Compute radiance... */
            obs->rad[id][ir] += src_planck[id] * eps * obs->tau[id][ir];
03241
03242
03243
            /\star Compute path transmittance... \star/
03244
            obs->tau[id][ir] *= (1 - eps);
03245
03246
03247
      03248
03249
03250
03251
          obs->rad[id][ir] += src_planck[id] * obs->tau[id][ir];
03252
03253
03254
      /* Free... */
03255
03256
      free(los);
03257 }
```

Here is the call graph for this function:



```
5.15.2.17 void formod_srcfunc ( ctl_t * ctl, tbl_t * tbl, double t, double * src )
```

Compute Planck source function.

Definition at line 3261 of file jurassic.c.

```
03265
                  {
03266
03267
      int id, it;
03268
03269
      /\star Determine index in temperature array... \star/
03270
      it = locate(tbl->st, TBLNS, t);
03271
03272
      /* Interpolate Planck function value... */
03273
      for (id = 0; id < ctl->nd; id++)
      03274
03275
03276 }
```

Here is the call graph for this function:



```
5.15.2.18 void geo2cart ( double z, double lon, double lat, double *x )
```

Convert geolocation to Cartesian coordinates.

Definition at line 3280 of file jurassic.c.

```
03284 {
03285
03286 double radius;
03287
03288 radius = z + RE;
03289 x[0] = radius * cos(lat / 180 * M_PI) * cos(lon / 180 * M_PI);
03290 x[1] = radius * cos(lat / 180 * M_PI) * sin(lon / 180 * M_PI);
03291 x[2] = radius * sin(lat / 180 * M_PI);
03292 }
```

5.15.2.19 double gravity (double z, double lat)

Determine gravity of Earth.

Definition at line 3296 of file jurassic.c.

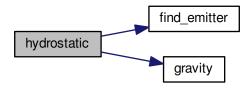
```
5.15.2.20 void hydrostatic ( ctl_t * ctl, atm_t * atm )
```

Set hydrostatic equilibrium.

Definition at line 3308 of file jurassic.c.

```
03310
03311
03312
         static int ig_h2o = -999;
03313
03314
         double dzmin = 1e99, e = 0, mean, mmair = 28.96456e-3, mmh2o =
03315
           18.0153e-3, z;
03316
03317
         int i, ip, ipref = 0, ipts = 20;
03318
         /* Check reference height... */
03320
         if (ctl->hydz < 0)
03321
            return;
03322
03323
         /* Determine emitter index of H2O... */
03324
         if (ig_h2o == -999)
03325
           ig_h2o = find_emitter(ctl, "H2O");
03326
03327
          /\star Find air parcel next to reference height... \star/
         for (ip = 0; ip < atm->np; ip++)
  if (fabs(atm->z[ip] - ctl->hydz) < dzmin) {
    dzmin = fabs(atm->z[ip] - ctl->hydz);
03328
03329
03330
              ipref = ip;
03331
03332
03333
         /* Upper part of profile... */
for (ip = ipref + 1; ip < atm->np; ip++) {
03334
03335
03336
           mean = 0;
            for (i = 0; i < ipts; i++) {</pre>
03337
03338
              z = LIN(0.0, atm \rightarrow z[ip - 1], ipts - 1.0, atm \rightarrow z[ip], (double) i);
03339
              if (ig_h2o >= 0)
03340
                e = LIN(0.0, atm->q[ig_h2o][ip - 1],
03341
              ipts - 1.0, atm->q[ig_h2o][ip], (double) i); mean += (e * mmh2o + (1 - e) * mmair)
03342
03343
                 * gravity(z, atm->lat[ipref]) / GSL_CONST_MKSA_MOLAR_GAS
03344
                 / LIN(0.0, atm->t[ip - 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03345
03346
           /* Compute p(z,T)... */
03347
03348
           atm->p[ip] =
03349
              \exp(\log(\arctan - p[ip - 1]) - mean * 1000 * (atm - z[ip] - atm - z[ip - 1]));
03350
03351
03352
         /\star Lower part of profile... \star/
03353
         for (ip = ipref - 1; ip >= 0; ip--) {
03354
           mean = 0;
            for (i = 0; i < ipts; i++) {</pre>
03355
03356
              z = LIN(0.0, atm \rightarrow z[ip + 1], ipts - 1.0, atm \rightarrow z[ip], (double) i);
03357
              if (ig_h2o >= 0)
03358
                e = LIN(0.0, atm->q[ig_h2o][ip + 1],
              ipts - 1.0, atm->q[ig_h2o][ip], (double) i);
mean += (e * mmh2o + (1 - e) * mmair)
  * gravity(z, atm->lat[ipref]) / GSL_CONST_MKSA_MOLAR_GAS
03359
03360
03361
03362
                 / LIN(0.0, atm->t[ip + 1], ipts - 1.0, atm->t[ip], (double) i) / ipts;
03363
03364
03365
            /* Compute p(z,T) \dots */
03366
           atm->p[ip]
03367
              \exp(\log(\text{atm->p[ip + 1]}) - \text{mean} * 1000 * (\text{atm->z[ip] - atm->z[ip + 1]}));
03368
03369 }
```

Here is the call graph for this function:



```
5.15.2.21 void idx2name ( ctl t * ctl, int idx, char * quantity )
```

Determine name of state vector quantity for given index.

Definition at line 3373 of file jurassic.c.

```
03376
03378
          int ig, iw;
03379
03380
          if (idx == IDXP)
             sprintf(quantity, "PRESSURE");
03381
03382
          if (idx == IDXT)
03383
03384
             sprintf(quantity, "TEMPERATURE");
03385
          for (ig = 0; ig < ctl->ng; ig++)
  if (idx == IDXQ(ig))
    sprintf(quantity, "%s", ctl->emitter[ig]);
03386
03387
03388
03389
          for (iw = 0; iw < ctl->nw; iw++)
  if (idx == IDXK(iw))
    sprintf(quantity, "EXTINCT_WINDOW%d", iw);
03390
03391
03392
03393 }
```

```
5.15.2.22 void init_tbl ( ctl_t * ctl, tbl_t * tbl )
```

Initialize look-up tables.

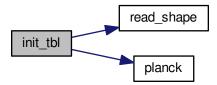
Definition at line 3397 of file jurassic.c.

```
03399
                         {
03400
03401
         FILE *in;
03402
         char filename[LEN], line[LEN];
03403
03404
         double eps, eps_old, press, press_old, temp, temp_old, u, u_old,
   f[NSHAPE], fsum, nu[NSHAPE];
03405
03406
03407
03408
         int i, id, ig, ip, it, n;
03409
         /* Loop over trace gases and channels... */
03410
03411 for (ig = 0; ig < ctl->ng; ig++)
03412 #pragma omp parallel for default (none) shared(ctl,tbl,ig) private(in,filename,line,eps,eps_old,press,
      press_old, temp, temp_old, u, u_old, id, ip, it)
03413
           for (id = 0; id < ctl->nd; id++) {
03414
03415
              /* Initialize... */
              tbl->np[ig][id] = -1;
eps_old = -999;
03416
03417
```

```
03418
             press_old = -999;
             temp\_old = -999;
03419
03420
             u_old = -999;
03421
             03422
03423
03424
03425
             if (!(in = fopen(filename, "r"))) {
03426
              printf("Missing emissivity table: %s\n", filename);
03427
                continue;
03428
             printf("Read emissivity table: %s\n", filename);
03429
03430
03431
              /* Read data... */
03432
             while (fgets(line, LEN, in)) {
03433
                /* Parse line... */ if (sscanf(line, "%lg %lg %lg %lg", &press, &temp, &u, &eps) != 4)
03434
03435
03436
                  continue;
03437
03438
                /* Determine pressure index... */
                if (press != press_old) {
  press_old = press;
  if ((++tbl->np[ig][id]) >= TBLNP)
03439
03440
03441
                  ERRMSG("Too many pressure levels!");
tbl->nt[ig][id][tbl->np[ig][id]] = -1;
03442
03443
03444
03445
03446
                /\star Determine temperature index... \star/
                if (temp != temp_old) {
  temp_old = temp;
03447
03448
03449
                     ((++tbl->nt[ig][id][tbl->np[ig][id]]) >= TBLNT)
03450
                    ERRMSG("Too many temperatures!");
03451
                  tbl->nu[ig][id][tbl->np[ig][id]]
03452
                    [tbl->nt[ig][id][tbl->np[ig][id]]] = -1;
03453
03454
03455
                /* Determine column density index... */
03456
                if ((eps > eps_old && u > u_old) || tbl->nu[ig][id][tbl->np[ig][id]]
03457
                    [tbl->nt[ig][id][tbl->np[ig][id]]] < 0) {
03458
                  eps_old = eps;
                  u\_old = u;
03459
                 if ((++tbl->nu[ig][id][tbl->np[ig][id]]
03460
                        [tbl->nt[ig][id][tbl->np[ig][id]]]) >= TBLNU) {
03461
                    tbl->nu[ig][id][tbl->np[ig][id]]
03462
03463
                      [tbl->nt[ig][id][tbl->np[ig][id]]]--;
03464
                    continue;
03465
                 }
               }
03466
03467
03468
                /* Store data... */
03469
                tbl->p[ig][id][tbl->np[ig][id]] = press;
03470
                \label{tbl->t[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]} tbl->t[ig][id][tbl->np[ig][id]]]
03471
                  = temp;
                tbl->u[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
[tbl->nu[ig][id][tbl->np[ig][id]]
[tbl->nt[ig][id][tbl->np[ig][id]]] = (float) u;
03472
03473
03474
03475
                tbl->eps[ig][id][tbl->np[ig][id]][tbl->nt[ig][id][tbl->np[ig][id]]]
03476
                  [tbl->nu[ig][id][tbl->np[ig][id]]
03477
                   [tbl->nt[ig][id][tbl->np[ig][id]]] = (float) eps;
03478
03479
03480
              /* Increment counters... */
03481
             tbl->np[ig][id]++;
03482
             for (ip = 0; ip < tbl->np[ig][id]; ip++) {
03483
                tbl->nt[ig][id][ip]++;
                for (it = 0; it < tbl->nt[ig][id][ip]; it++)
03484
                  tbl->nu[ig][id][ip][it]++;
03485
03486
03487
03488
             /* Close file... */
03489
             fclose(in);
03490
          }
03491
03492
        /* Write info... */
        printf("Initialize source function table...\n");
03493
03494
03495
        /* Loop over channels... */
03496 #pragma omp parallel for default(none) shared(ctl,tbl,ig) private(filename,it,i,n,f,fsum,nu) 03497 for (id = 0; id < ctl->nd; id++) {
03498
03499
           /* Read filter function... */
03500
           sprintf(filename, "%s_%.4f.filt", ctl->tblbase, ctl->nu[id]);
03501
           read_shape(filename, nu, f, &n);
03502
           /* Compute source function table... */
for (it = 0; it < TBLNS; it++) {</pre>
03503
03504
```

```
/* Set temperature... */ tbl->st[it] = LIN(0.0, TMIN, TBLNS - 1.0, TMAX, (double) it);
03506
03507
03508
03509
             /* Integrate Planck function... */
03510
             fsum = 0;
03511
             tbl->sr[id][it] = 0;
03512
             for (i = 0; i < n; i++) {</pre>
03513
              fsum += f[i];
               tbl->sr[id][it] += f[i] * planck(tbl->st[it], nu[i]);
03514
03515
03516
             tbl->sr[id][it] /= fsum;
03517
03518
03519 }
```

Here is the call graph for this function:



5.15.2.23 void intpol\_atm ( ctl\_t \* ctl, atm\_t \* atm, double z, double \* p, double \* t, double \* q, double \* k)

Interpolate atmospheric data.

Definition at line 3523 of file jurassic.c.

```
03530
                          {
03531
03532
          int ig, ip, iw;
03533
03534
          /* Get array index... */
03535
          ip = locate(atm->z, atm->np, z);
03536
          /* Interpolate... */
          *p = EXP(atm->z[ip], atm->p[ip], atm->z[ip + 1], atm->p[ip + 1], z);
*t = LIN(atm->z[ip], atm->t[ip], atm->z[ip + 1], atm->t[ip + 1], z);
03538
03539
          for (ig = 0; ig < ctl->ng; ig++)
  q[ig] =
03540
03541
          LIN(atm->z[ip], atm->q[ig][ip], atm->z[ip + 1], atm->q[ig][ip + 1], z); for (iw = 0; iw < ctl->nw; iw++)
03542
03543
03544
03545
                \label{eq:linear} LIN\,(atm->z\,[ip],\ atm->k\,[iw]\,[ip],\ atm->z\,[ip+1],\ atm->k\,[iw]\,[ip+1],\ z)\,;
03546 }
```

Here is the call graph for this function:



5.15.2.24 void intpol\_tbl ( ctl\_t \* ctl, tbl\_t \* tbl, los\_t \* los, int ip, double tau\_path[NG][ND], double tau\_seg[ND] )

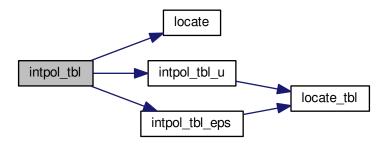
Get transmittance from look-up tables.

Definition at line 3550 of file jurassic.c.

```
03556
03557
03558
        double eps, eps00, eps01, eps10, eps11, u;
03559
03560
        int id, ig, ipr, it0, it1;
03561
03562
        /* Initialize... */
03563
        if (ip <= 0)</pre>
03564
         for (ig = 0; ig < ctl->ng; ig++)
03565
            for (id = 0; id < ctl->nd; id++)
03566
              tau_path[ig][id] = 1;
03567
03568
        /* Loop over channels... */
03569
        for (id = 0; id < ctl->nd; id++) {
03570
           /* Initialize... */
03571
03572
          tau_seg[id] = 1;
03573
03574
           /* Loop over emitters.... */
03575
          for (ig = 0; ig < ctl->ng; ig++) {
03576
03577
             /\star Check size of table (pressure)... \star/
03578
             if (tbl->np[ig][id] < 2)</pre>
03579
              eps = 0;
03580
03581
             /* Check transmittance... */
             else if (tau_path[ig][id] < 1e-9)</pre>
03583
              eps = 1;
03584
03585
             /* Interpolate... */
03586
             else {
03587
03588
               /* Determine pressure and temperature indices... */
               ipr = locate(tbl->p[ig][id], tbl->np[ig][id], los->p[ip]);
it0 = locate(tbl->t[ig][id][ipr], tbl->nt[ig][id][ipr], los->t[ip]);
03589
03590
               it1 =
03591
03592
                 locate(tbl->t[ig][id][ipr + 1], tbl->nt[ig][id][ipr + 1],
03593
                         los->t[ip]);
03594
03595
               /\star Check size of table (temperature and column density)... \star/
03596
               if (tbl->nt[ig][id][ipr] < 2 || tbl->nt[ig][id][ipr + 1] < 2</pre>
                   || tbl->nu[ig][id][ipr][it0] < 2
|| tbl->nu[ig][id][ipr][it0 + 1] < 2
03597
03598
03599
                   || tbl->nu[ig][id][ipr + 1][it1] < 2
|| tbl->nu[ig][id][ipr + 1][it1 + 1] < 2)
03600
03601
                 eps = 0;
03602
03603
               else {
03604
                 /* Get emissivities of extended path... */
u = intpol_tbl_u(tbl, ig, id, ipr, it0, 1 - tau_path[ig][id]);
eps00 = intpol_tbl_eps(tbl, ig, id, ipr, it0, u + los->u[ig][ip]);
03605
03606
03607
03608
03609
                 u = intpol_tbl_u(tbl, ig, id, ipr, it0 + 1, 1 - tau_path[ig][id]);
03610
                 eps01 =
                   intpol_tbl_eps(tbl, ig, id, ipr, it0 + 1, u + los->u[ig][ip]);
03611
03612
03613
                 u = intpol_tbl_u(tbl, ig, id, ipr + 1, it1, 1 - tau_path[ig][id]);
03614
                 eps10 =
03615
                   intpol_tbl_eps(tbl, ig, id, ipr + 1, it1, u + los->u[ig][ip]);
03616
03617
03618
                   intpol tbl u(tbl, ig, id, ipr + 1, it1 + 1, 1 - tau path[ig][id]);
03619
                 eps11
                   intpol_tbl_eps(tbl, ig, id, ipr + 1, it1 + 1, u + los->
      u[ig][ip]);
03621
03622
                 /* Interpolate with respect to temperature... */
                 03623
03624
                 eps11 = LIN(tbl->t[ig][id][ipr + 1][it1], eps10,
03625
03626
                              tbl->t[ig][id][ipr + 1][it1 + 1], eps11, los->t[ip]);
03627
03628
                 /* Interpolate with respect to pressure... */
                 03629
03630
03631
```

```
/* Check emssivity range... */
03633
                  eps00 = GSL_MAX(GSL_MIN(eps00, 1), 0);
03634
                  /* Determine segment emissivity... */
eps = 1 - (1 - eps00) / tau_path[ig][id];
03635
03636
                }
03637
03638
03639
03640
              /\star Get transmittance of extended path... \star/
03641
              tau_path[ig][id] *= (1 - eps);
03642
03643
              /* Get segment transmittance... */
03644
             tau_seg[id] *= (1 - eps);
03645
03646
03647 }
```

Here is the call graph for this function:



5.15.2.25 double intpol\_tbl\_eps ( tbl\_t \* tbl, int ig, int id, int ip, int it, double u )

Interpolate emissivity from look-up tables.

Definition at line 3651 of file jurassic.c.

```
03657
                    {
03658
03659
         int idx;
03660
03661
         /* Lower boundary... */
03662
         if (u < tbl->u[ig][id][ip][it][0])
          return LIN(0, 0, tbl->u[ig][id][ip][it][0], tbl->eps[ig][id][ip][it][0],
03663
03664
                       u);
03665
03666
         /* Upper boundary... */
         else if (u > tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1])
03667
           return LIN(tbl->u[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03668
                        tbl->eps[ig][id][ip][it][tbl->nu[ig][id][ip][it] - 1],
03669
03670
                        1e30, 1, u);
03671
03672
         /* Interpolation... */
03673
         else {
03674
           /* Get index... */  idx = locate\_tbl(tbl->u[ig][id][ip][it], \ tbl->nu[ig][id][ip][it], \ u); 
03675
03676
03677
03678
           /* Interpolate... */
03679
             LIN(tbl->u[ig][id][ip][it][idx], tbl->eps[ig][id][ip][it][idx], tbl->u[ig][id][ip][it][idx + 1], tbl->eps[ig][id][ip][it][idx + 1],
03680
03681
03682
                  u);
03683
         }
03684 }
```

Here is the call graph for this function:



```
5.15.2.26 double intpol_tbl_u ( tbl_t*tbl, int ig, int id, int ip, int it, double eps )
```

Interpolate column density from look-up tables.

Definition at line 3688 of file jurassic.c.

```
03694
03696
      int idx;
03697
      /* Lower boundary... */
if (eps < tbl->eps[ig][id][ip][it][0])
  return LIN(0, 0, tbl->eps[ig][id][ip][it][0], tbl->u[ig][id][ip][it][0],
03698
03699
03700
03701
                  eps);
03702
      03703
03704
03705
03706
                  1, 1e30, eps);
03708
03709
      /* Interpolation... */
03710
      else {
0.3711
03712
        /* Get index... */
03713
        idx = locate_tbl(tbl->eps[ig][id][ip][it], tbl->nu[ig][id][ip][it], eps);
03714
03715
03716
        return
03717
         03718
03719
             eps);
03720
03721 }
```

Here is the call graph for this function:



5.15.2.27 void jsec2time ( double jsec, int \* year, int \* mon, int \* day, int \* hour, int \* min, int \* sec, double \* remain )

Convert seconds to date.

Definition at line 3725 of file jurassic.c.

```
03733
03734
03735
        struct tm t0, *t1;
03737
        time_t jsec0;
03738
03739
        t0.tm\_year = 100;
        t0.tm_mon = 0;
03740
03741
        t0.tm_mday = 1;
03742
        t0.tm\_hour = 0;
03743
        t0.tm_min = 0;
        t0.tm_sec = 0;
03744
03745
03746
        jsec0 = (time_t) jsec + timegm(&t0);
03747
        t1 = gmtime(&jsec0);
03748
03749
        *year = t1->tm_year + 1900;
03750
        *mon = t1->tm_mon + 1;
        *day = t1->tm_mday;
03751
        *hour = t1->tm_hour;
03752
03753
        *min = t1->tm_min;
        *sec = t1->tm_sec;
*remain = jsec - floor(jsec);
03754
03755
03756 }
```

5.15.2.28 void kernel (  $ctl_t * ctl$ ,  $atm_t * atm$ ,  $obs_t * obs$ ,  $gsl_matrix * k$  )

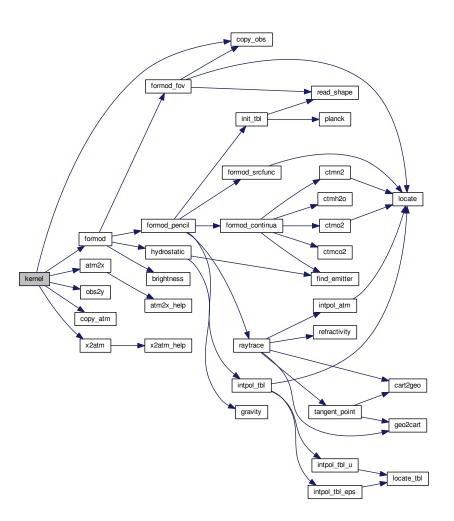
Compute Jacobians.

Definition at line 3760 of file jurassic.c.

```
03764
                                                                              {
03765
03766
                         atm_t *atm1;
03767
                        obs_t *obs1;
03768
03769
                        gsl_vector *x0, *x1, *yy0, *yy1;
03770
03771
                         int *iqa, j;
03772
03773
                        double h;
03774
03775
                        size_t i, n, m;
03776
03777
                        /* Get sizes... */
                        m = k->size1;
n = k->size2;
03778
03779
03780
03781
                         /* Allocate... */
03782
                        x0 = gsl\_vector\_alloc(n);
                         yy0 = gsl_vector_alloc(m);
03784
                         ALLOC(iqa, int,
03785
                                             N);
03786
03787
                         /* Compute radiance for undisturbed atmospheric data... */
03788
                         formod(ctl, atm, obs);
03789
03790
                         /* Compose vectors... */
03791
                         atm2x(ctl, atm, x0, iqa, NULL);
03792
                         obs2y(ctl, obs, yy0, NULL, NULL);
03793
03794
                        /* Initialize kernel matrix... */
03795
                        gsl_matrix_set_zero(k);
03796
03797
                          /\star Loop over state vector elements... \star/
03798 \text{ \#pragma omp parallel for default(none) shared(ctl,atm,obs,k,x0,yy0,n,m,iqa) private(i, j, h, x1, yy1, atm1, h, x1, yy1, h, x1, 
                      obs1)
03799
                         for (j = 0; j < (int) n; j++) {
03800
03801
                               /* Allocate... */
```

```
x1 = gsl_vector_alloc(n);
03803
           yy1 = gsl_vector_alloc(m);
03804
            ALLOC(atm1, atm_t, 1);
           ALLOC(obs1, obs_t, 1);
03805
03806
03807
            /* Set perturbation size... */
03808
           if (iqa[j] == IDXP)
03809
             h = GSL_MAX(fabs(0.01 * gsl_vector_get(x0, (size_t) j)), 1e-7);
03810
            else if (iqa[j] == IDXT)
             h = 1;
03811
           else if (iqa[j] >= IDXQ(0) && iqa[j] < IDXQ(ctl->ng))
03812
           \label{eq:heat_max} \begin{array}{ll} h = GSL\_MAX(fabs(0.01 * gsl\_vector\_get(x0, (size\_t) j)), \ 1e-15); \\ else \ if \ (iqa[j] >= IDXK(0) \ \&\& \ iqa[j] < IDXK(ctl->nw)) \end{array}
03813
03814
03815
03816
            else
             ERRMSG("Cannot set perturbation size!");
03817
03818
03819
            /* Disturb state vector element... */
03820
           gsl_vector_memcpy(x1, x0);
03821
           gsl_vector_set(x1, (size_t) j, gsl_vector_get(x1, (size_t) j) + h);
           copy_atm(ctl, atm1, atm, 0);
copy_obs(ctl, obs1, obs, 0);
03822
03823
03824
           x2atm(ctl, x1, atm1);
03825
03826
            /* Compute radiance for disturbed atmospheric data... */
           formod(ctl, atml, obsl);
03828
03829
            /\star Compose measurement vector for disturbed radiance data... \star/
03830
           obs2y(ctl, obs1, yy1, NULL, NULL);
03831
03832
            /* Compute derivatives... */
03833
            for (i = 0; i < m; i++)
03834
             gsl_matrix_set(k, i, (size_t) j,
03835
                               (gsl_vector_get(yy1, i) - gsl_vector_get(yy0, i)) / h);
03836
           /* Free... */
gsl_vector_free(x1);
gsl_vector_free(yy1);
03837
03838
03839
03840
            free(atm1);
03841
           free (obs1);
03842
03843
         /* Free... */
gsl_vector_free(x0);
03844
03845
03846
         gsl_vector_free(yy0);
03847
         free(iqa);
03848 }
```

Here is the call graph for this function:



5.15.2.29 int locate ( double \*xx, int n, double x )

Find array index.

Definition at line 3852 of file jurassic.c.

```
{
03856
          int i, ilo, ihi;
03857
03858
03859
          ilo = 0;
         ihi = n - 1;
i = (ihi + ilo) >> 1;
03860
03861
03862
          if (xx[i] < xx[i + 1])
while (ihi > ilo + 1) {
   i = (ihi + ilo) >> 1;
   if (xx[i] > x)
      ihi = i;
03863
03864
03865
03866
03867
               else
03868
03869
                 ilo = i;
         03870
03871
03872
03873
03874
                 ihi = i;
```

```
03875 else
03876 ilo = i;
03877 }
03878 03879 return ilo;
03880 }
```

5.15.2.30 int locate\_tbl (float \*xx, int n, double x)

Find array index in float array.

Definition at line 3884 of file jurassic.c.

```
03887
                  {
03888
03889
       int i, ilo, ihi;
03890
03891
       ilo = 0;
03892
       ihi = n - 1;
03893
       i = (ihi + ilo) >> 1;
03894
       while (ihi > ilo + 1) {
03895
        i = (ihi + ilo) >> 1;
if (xx[i] > x)
03896
03897
03898
            ihi = i;
03899
         else
03900
            ilo = i;
       }
03901
03902
03903
       return ilo;
03904 }
```

5.15.2.31 size\_t obs2y ( ctl\_t \* ctl, obs\_t \* obs, gsl\_vector \* y, int \* ida, int \* ira )

Compose measurement vector.

Definition at line 3908 of file jurassic.c.

```
03913
                    {
03914
03915
        int id, ir;
03916
03917
        size_t m = 0;
03918
03919
        /* Determine measurement vector... */
03920
        for (ir = 0; ir < obs->nr; ir++)
        for (id = 0; id < ctl->nd; id++)
03921
03922
            if (gsl_finite(obs->rad[id][ir])) {
             if (y != NULL)
   gsl_vector_set(y, m, obs->rad[id][ir]);
if (ida != NULL)
03923
03924
03925
              ida[m] = id;
if (ira != NULL)
03926
03927
03928
                 ira[m] = ir;
03929
              m++;
03930
03931
03932
       return m;
03933 }
```

5.15.2.32 double planck (double t, double nu)

Compute Planck function.

Definition at line 3937 of file jurassic.c.

```
5.15.2.33 void raytrace ( ctl_t * ctl, atm_t * atm, obs_t * obs, los_t * los, int ir )
```

Do ray-tracing to determine LOS.

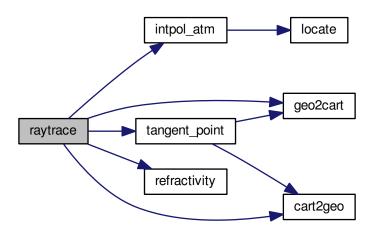
Definition at line 3946 of file jurassic.c.

```
03951
                 {
03952
03953
        double cosa, d, dmax, dmin = 0, ds, ex0[3], ex1[3], frac, h = 0.02, k[NW],
03954
         lat, lon, n, naux, ng[3], norm, p, q[NG], t, x[3], xh[3],
03955
          xobs[3], xvp[3], z = 1e99, zmax, zmin, zrefrac = 60;
03956
03957
        int i, ig, ip, iw, stop = 0;
03959
        /* Initialize... */
03960
        los->np = 0;
        los->tsurf = -999;
obs->tpz[ir] = obs->vpz[ir];
03961
03962
03963
        obs->tplon[ir] = obs->vplon[ir];
03964
        obs->tplat[ir] = obs->vplat[ir];
03965
03966
        /* Get altitude range of atmospheric data... */
03967
        gsl_stats_minmax(&zmin, &zmax, atm->z, 1, (size_t) atm->np);
03968
03969
        /* Check observer altitude... */
03970
           (obs->obsz[ir] < zmin)
03971
          ERRMSG("Observer below surface!");
03972
03973
        /* Check view point altitude... */
03974
        if (obs->vpz[ir] > zmax)
03975
         return;
03976
03977
        /* Determine Cartesian coordinates for observer and view point... */
03978
        geo2cart(obs->obsz[ir], obs->obslon[ir], obs->obslat[ir], xobs);
03979
        geo2cart(obs->vpz[ir], obs->vplon[ir], obs->vplat[ir], xvp);
03980
        /* Determine initial tangent vector... */
03981
03982
        for (i = 0; i < 3; i++)
03983
          ex0[i] = xvp[i] - xobs[i];
        norm = NORM(ex0);
for (i = 0; i < 3; i++)
03984
03985
03986
          ex0[i] /= norm;
03987
03988
        /* Observer within atmosphere... */
        for (i = 0; i < 3; i++)
03989
03990
          x[i] = xobs[i];
03991
03992
        /* Observer above atmosphere (search entry point)... */
03993
        if (obs->obsz[ir] > zmax) {
03994
          dmax = norm;
03995
          while (fabs(dmin - dmax) > 0.001) {
03996
            d = (dmax + dmin) / 2;
            for (i = 0; i < 3; i++)
x[i] = xobs[i] + d * ex0[i];
03997
03998
03999
            cart2geo(x, &z, &lon, &lat);
            if (z \le zmax && z > zmax - 0.001)
04000
04001
              break;
04002
             if (z < zmax - 0.0005)
04003
              dmax = d;
04004
            else
              dmin = d;
04005
04006
          }
04007
        }
04008
04009
        /* Ray-tracing... */
04010
        while (1) {
04011
04012
          /* Set step length... */
04013
          ds = ctl->ravds;
          if (ctl->raydz > 0) {
04014
04015
            norm = NORM(x);
            for (i = 0; i < 3; i++)
   xh[i] = x[i] / norm;</pre>
04016
04017
             cosa = fabs(DOTP(ex0, xh));
04018
04019
            if (cosa != 0)
04020
              ds = GSL_MIN(ctl->rayds, ctl->raydz / cosa);
04021
04022
04023
          /* Determine geolocation... */
04024
          cart2geo(x, &z, &lon, &lat);
04025
04026
          /* Check if LOS hits the ground or has left atmosphere... */
04027
          if (z < zmin || z > zmax) {
```

```
04028
              stop = (z < zmin ? 2 : 1);
04029
               ((z <
04030
04031
                  zmin ? zmin : zmax) - los-z[los-np - 1]) / (z - los-z[los-np - 1])
04032
                                                                                      11);
              geo2cart(los->z[los->np - 1], los->lon[los->np - 1],
04033
              los->lat[los->np - 1], xh);
for (i = 0; i < 3; i++)
04034
04035
04036
               x[i] = xh[i] + frac * (x[i] - xh[i]);
04037
              cart2geo(x, &z, &lon, &lat);
              los->ds[los->np - 1] = ds * frac;
04038
04039
              ds = 0:
04040
04041
04042
            /* Interpolate atmospheric data... */
04043
           intpol_atm(ctl, atm, z, &p, &t, q, k);
04044
04045
            /* Save data... */
           los -> lon[los -> np] = lon;
04046
04047
           los->lat[los->np] = lat;
04048
           los \rightarrow z[los \rightarrow np] = z;
04049
           los \rightarrow p[los \rightarrow np] = p;
           los->t[los->np] = t;
04050
           for (ig = 0; ig < ctl->ng; ig++)
los->q[ig][los->np] = q[ig];
04051
04052
04053
           for (iw = 0; iw < ctl->nw; iw++)
04054
             los \rightarrow k[iw][los \rightarrow np] = k[iw];
04055
           los->ds[los->np] = ds;
04056
04057
           /* Increment and check number of LOS points... */
04058
           if ((++los->np) > NLOS)
04059
              ERRMSG("Too many LOS points!");
04060
04061
            /* Check stop flag... */
04062
           if (stop) {
             los->tsurf = (stop == 2 ? t : -999);
04063
04064
             break;
04065
04066
04067
            /* Determine refractivity... */
04068
           if (ctl->refrac && z <= zrefrac)</pre>
04069
             n = 1 + refractivity(p, t);
04070
           else
04071
             n = 1;
04072
04073
           /\star Construct new tangent vector (first term)... \star/
04074
           for (i = 0; i < 3; i++)
              ex1[i] = ex0[i] * n;
04075
04076
04077
           /* Compute gradient of refractivity... */
           if (ctl->refrac && z <= zrefrac) {
             for (i = 0; i < 3; i++)
xh[i] = x[i] + 0.5 * ds * ex0[i];
04079
04080
              cart2geo(xh, &z, &lon, &lat);
intpol_atm(ctl, atm, z, &p, &t, q, k);
n = refractivity(p, t);
for (i = 0; i < 3; i++) {</pre>
04081
04082
04083
04084
                xh[i] += h;
04085
04086
                cart2geo(xh, &z, &lon, &lat);
                intpol_atm(ctl, atm, z, &p, &t, q, k);
naux = refractivity(p, t);
04087
04088
                ng[i] = (naux - n) / h;
04089
04090
                xh[i] -= h;
04091
04092
           } else
04093
              for (i = 0; i < 3; i++)
               ng[i] = 0;
04094
04095
04096
           /* Construct new tangent vector (second term)... */
           for (i = 0; i < 3; i++)
04097
04098
             ex1[i] += ds * ng[i];
04099
04100
           /\star Normalize new tangent vector... \star/
04101
           norm = NORM(ex1);
           for (i = 0; i < 3; i++)
ex1[i] /= norm;
04102
04103
04104
04105
            /* Determine next point of LOS... */
           for (i = 0; i < 3; i++)
x[i] += 0.5 * ds * (ex0[i] + ex1[i]);</pre>
04106
04107
04108
04109
           /* Copy tangent vector... */
for (i = 0; i < 3; i++)</pre>
04110
04111
              ex0[i] = ex1[i];
04112
04113
04114
         /* Get tangent point (to be done before changing segment lengths!)... */
```

```
04115
       tangent_point(los, &obs->tpz[ir], &obs->tplon[ir], &obs->
04116
04117
       /\star Change segment lengths according to trapezoid rule... \star/
       for (ip = los->np - 1; ip >= 1; ip--)
los->ds[ip] = 0.5 * (los->ds[ip - 1] + los->ds[ip]);
04118
04119
04120
       los->ds[0] *= 0.5;
04121
04122
       /* Compute column density... */
       04123
04124
04125
04126
04127 }
```

Here is the call graph for this function:



5.15.2.34 void read\_atm ( const char \* dirname, const char \* filename, ctl t \* ctl, atm t \* atm )

Read atmospheric data.

Definition at line 4131 of file jurassic.c.

```
04135
04136
04137
        FILE *in;
04138
04139
        char file[LEN], line[LEN], *tok;
04140
04141
       int ig, iw;
04142
04143
       /* Init... */
04144
       atm->np = 0;
04145
04146
        /* Set filename... */
        if (dirname != NULL)
04147
04148
         sprintf(file, "%s/%s", dirname, filename);
04149
04150
         sprintf(file, "%s", filename);
04151
04152
        /* Write info... */
       printf("Read atmospheric data: %s\n", file);
04153
04154
04155
        /* Open file... */
04156
       if (!(in = fopen(file, "r")))
04157
         ERRMSG("Cannot open file!");
```

```
04158
04159
                /* Read line... */
04160
                while (fgets(line, LEN, in)) {
04161
                  /* Read data... */
TOK(line, tok, "%lg", atm->time[atm->np]);
TOK(NULL, tok, "%lg", atm->z[atm->np]);
TOK(NULL, tok, "%lg", atm->z[atm->np]);
TOK(NULL, tok, "%lg", atm->lat[atm->np]);
TOK(NULL, tok, "%lg", atm->p[atm->np]);
TOK(NULL, tok, "%lg", atm->t[atm->np]);
TOK(NULL, tok, "%lg", atm->t[atm->np]);
for (ig = 0; ig < ctl->ng; ig++)
TOK(NULL, tok, "%lg", atm->q[ig][atm->np]);
for (iw = 0; iw < ctl->nw; iw++)
TOK(NULL, tok, "%lg", atm->k[iw][atm->np]);
04162
04163
04164
04165
04166
04167
04168
04169
04170
04171
04172
04173
                 /* Increment data point counter... */
if ((++atm->np) > NP)
04174
04175
                       ERRMSG("Too many data points!");
04177
04178
04179
               /* Close file... */
04180
               fclose(in);
04181
04182
                /* Check number of points... */
04183
                if (atm->np < 1)
04184
                    ERRMSG("Could not read any data!");
04185 }
```

5.15.2.35 void read\_ctl ( int argc, char \* argv[], ctl\_t \* ctl )

Read forward model control parameters.

Definition at line 4189 of file jurassic.c.

```
04192
04193
04194
         int id, ig, iw;
04195
04196
         /* Write info... */
         printf("\nJuelich Rapid Spectral Simulation Code (JURASSIC)\n"
04197
                   "(executable: %s | compiled: %s, %s)\n\n",
04198
04199
                   argv[0], __DATE__, __TIME__);
04200
04201
         /* Emitters... */
         ctl->ng = (int) scan_ctl(argc, argv, "NG", -1, "0", NULL); if (ctl->ng < 0 || ctl->ng > NG)
04202
04203
            ERRMSG("Set 0 <= NG <= MAX!");
04204
04205
          for (ig = 0; ig < ctl->ng; ig++)
04206
            scan_ctl(argc, argv, "EMITTER", ig, "", ctl->emitter[ig]);
04207
         /* Radiance channels... */
ctl->nd = (int) scan_ctl(argc, argv, "ND", -1, "0", NULL);
if (ctl->nd < 0 || ctl->nd > ND)
04208
04209
04210
04211
            ERRMSG("Set 0 <= ND <= MAX!");</pre>
04212
          for (id = 0; id < ctl->nd; id++)
04213
            ctl->nu[id] = scan_ctl(argc, argv, "NU", id, "", NULL);
04214
04215
          /* Spectral windows... */
04216
         ctl->nw = (int) scan_ctl(argc, argv, "NW", -1, "1", NULL);
04217
          if (ctl->nw < 0 || ctl->nw > NW)
04218
            ERRMSG("Set 0 <= NW <= MAX!");</pre>
04219
          for (id = 0; id < ctl->nd; id++)
04220
            ctl->window[id] = (int) scan_ctl(argc, argv, "WINDOW", id, "0", NULL);
04221
         /* Emissivity look-up tables... */
scan_ctl(argc, argv, "TBLBASE", -1, "-", ctl->tblbase);
04222
04223
04224
04225
          /* Hydrostatic equilibrium... */
          ctl->hydz = scan_ctl(argc, argv, "HYDZ", -1, "-999", NULL);
04226
04227
04228
         /* Continua... */
         ctl->ctm_co2 = (int) scan_ctl(argc, argv, "CTM_CO2", -1, "1", NULL);
ctl->ctm_h2o = (int) scan_ctl(argc, argv, "CTM_H2O", -1, "1", NULL);
ctl->ctm_n2 = (int) scan_ctl(argc, argv, "CTM_N2", -1, "1", NULL);
ctl->ctm_o2 = (int) scan_ctl(argc, argv, "CTM_O2", -1, "1", NULL);
04229
04230
04231
04232
04233
04234
          /* Ray-tracing... */
         ctl->refrac = (int) scan_ctl(argc, argv, "REFRAC", -1, "1", NULL);
ctl->rayds = scan_ctl(argc, argv, "RAYDS", -1, "10", NULL);
04235
```

```
ctl->raydz = scan_ctl(argc, argv, "RAYDZ", -1, "0.5", NULL);
04238
           /* Field of view... */
04239
           scan_ctl(argc, argv, "FOV", -1, "-", ctl->fov);
04240
04241
04242
           /* Retrieval interface... */
           ctl->retp_zmin = scan_ctl(argc, argv, "RETP_ZMIN", -1, "-999", NULL);
ctl->retp_zmax = scan_ctl(argc, argv, "RETP_ZMAX", -1, "-999", NULL);
ctl->rett_zmin = scan_ctl(argc, argv, "RETT_ZMIN", -1, "-999", NULL);
04243
04244
04245
           ctl->rett_zmax = scan_ctl(argc, argv, "RETT_ZMAX", -1, "-999", NULL);
for (ig = 0; ig < ctl->ng; ig++) {
04246
04247
            ctl->retq_zmin[ig] = scan_ctl(argc, argv, "RETO_ZMIN", ig, "-999", NULL); ctl->retq_zmax[ig] = scan_ctl(argc, argv, "RETO_ZMAX", ig, "-999", NULL);
04248
04249
04250
04251
           for (iw = 0; iw < ctl->nw; iw++) {
           ctl->retk_zmin[iw] = scan_ctl(argc, argv, "RETK_ZMIN", iw, "-999", NULL);
ctl->retk_zmax[iw] = scan_ctl(argc, argv, "RETK_ZMAX", iw, "-999", NULL);
04252
04253
04254
04256
           /* Output flags... */
04257
           ctl->write_bbt = (int) scan_ctl(argc, argv, "WRITE_BBT", -1, "0", NULL);
04258
           ctl->write_matrix =
              (int) scan_ctl(argc, argv, "WRITE_MATRIX", -1, "0", NULL);
04259
04260 }
```

Here is the call graph for this function:



5.15.2.36 void read\_matrix ( const char \* dirname, const char \* filename, gsl\_matrix \* matrix )

Read matrix.

Definition at line 4264 of file jurassic.c.

```
04267
04268
04269
       FILE *in;
04270
       char dum[LEN], file[LEN], line[LEN];
04271
04272
04273
       double value;
04274
04275
       int i, j;
04276
04277
        /* Set filename... */
04278
        if (dirname != NULL)
04279
         sprintf(file, "%s/%s", dirname, filename);
04280
04281
         sprintf(file, "%s", filename);
04282
04283
       /* Write info... */
04284
       printf("Read matrix: %s\n", file);
04285
04286
04287
       if (!(in = fopen(file, "r")))
04288
         ERRMSG("Cannot open file!");
04289
04290
       /* Read data... */
04291
       gsl_matrix_set_zero(matrix);
04292
       while (fgets(line, LEN, in))
04293
         if (sscanf(line, "%d %s %s %s %s %s %d %s %s %s %s %s %lg",
04294
                    &i, dum, dum, dum, dum, dum,
                     &j, dum, dum, dum, dum, &value) == 13)
04295
04296
            gsl_matrix_set(matrix, (size_t) i, (size_t) j, value);
04297
04298
        /* Close file... */
04299
       fclose(in);
04300 }
```

5.15.2.37 void read\_obs ( const char \* dirname, const char \* filename, ctl\_t \* ctl, obs\_t \* obs )

Read observation data.

Definition at line 4304 of file jurassic.c.

```
04308
04309
04310
             FILE *in;
04312
             char file[LEN], line[LEN], *tok;
04313
04314
             int id;
04315
             /* Init... */
obs->nr = 0;
04316
04317
04318
04319
              /* Set filename... */
04320
              if (dirname != NULL)
                sprintf(file, "%s/%s", dirname, filename);
04321
04322
             else
04323
                 sprintf(file, "%s", filename);
04324
04325
              /\star Write info... \star/
04326
             printf("Read observation data: %s\n", file);
04327
04328
             /* Open file... */
if (!(in = fopen(file, "r")))
04329
04330
                 ERRMSG("Cannot open file!");
04331
04332
              /* Read line... */
             while (fgets(line, LEN, in)) {
04333
                /* Read data... */
TOK(line, tok, "%lg", obs->time[obs->nr]);
TOK(NULL, tok, "%lg", obs->obsz[obs->nr]);
TOK(NULL, tok, "%lg", obs->obslon[obs->nr]);
TOK(NULL, tok, "%lg", obs->obslat[obs->nr]);
TOK(NULL, tok, "%lg", obs->vpz[obs->nr]);
TOK(NULL, tok, "%lg", obs->vplon[obs->nr]);
TOK(NULL, tok, "%lg", obs->vplat[obs->nr]);
TOK(NULL, tok, "%lg", obs->tpz[obs->nr]);
TOK(NULL, tok, "%lg", obs->tplon[obs->nr]);
TOK(NULL, tok, "%lg", obs->tplon[obs->nr]);
TOK(NULL, tok, "%lg", obs->tplat[obs->nr]);
for (id = 0; id < ctl->nd; id++)
    TOK(NULL, tok, "%lg", obs->rad[id][obs->nr]);
for (id = 0; id < ctl->nd; id++)
    TOK(NULL, tok, "%lg", obs->tplat[id][obs->nr]);
04334
04335
04336
04337
04338
04339
04340
04341
04342
04343
04344
04345
04346
04347
04348
04349
04350
                 /* Increment counter... */
04351
04352
                 if ((++obs->nr) > NR)
                     ERRMSG("Too many rays!");
04353
04354
04355
04356
              /* Close file... */
04357
             fclose(in);
04358
04359
             /* Check number of points... */
if (obs->nr < 1)</pre>
04360
                 ERRMSG("Could not read any data!");
04361
04362 }
```

5.15.2.38 void read\_shape ( const char \* filename, double \* x, double \* y, int \* n )

Read shape function.

Definition at line 4366 of file jurassic.c.

```
04370 {
04371
04372 FILE *in;
04373
04374 char line[LEN];
04375
04376 /* Write info... */
04377 printf("Read shape function: %s\n", filename);
```

```
04378
04379
         /* Open file... */
         if (!(in = fopen(filename, "r")))
04380
          ERRMSG("Cannot open file!");
04381
04382
         /* Read data... */
04383
04384
        *n = 0;
04385
        while (fgets(line, LEN, in))
         if (sscanf(line, "%lg %lg", &x[*n], &y[*n]) == 2)
if ((++(*n)) > NSHAPE)
    ERRMSG("Too many data points!");
04386
04387
04388
04389
04390
        /* Check number of points... */
04391
04392
          ERRMSG("Could not read any data!");
04393
        /* Close file... */
04394
04395
        fclose(in);
04396 }
```

5.15.2.39 double refractivity ( double p, double t )

Compute refractivity (return value is n - 1).

Definition at line 4400 of file jurassic.c.

5.15.2.40 double scan\_ctl (int argc, char \* argv[], const char \* varname, int arridx, const char \* defvalue, char \* value)

Search control parameter file for variable entry.

Definition at line 4410 of file jurassic.c.

```
04416
                            {
04417
04418
          FILE *in = NULL;
04419
04420
          char dummy[LEN], fullname1[LEN], fullname2[LEN], line[LEN],
04421
            msg[LEN], rvarname[LEN], rval[LEN];
04422
04423
          int contain = 0, i;
04424
04425
          /* Open file... */
          if (argv[1][0] != '-')
04426
           if (!(in = fopen(argv[1], "r")))
    ERRMSG("Cannot open file!");
04427
04428
04429
04430
          /* Set full variable name... */
04431
          if (arridx >= 0) {
           sprintf(fullname1, "%s[%d]", varname, arridx);
sprintf(fullname2, "%s[*]", varname);
04432
04433
04434
          sprintf(fullname1, "%s", varname);
sprintf(fullname2, "%s", varname);
04435
04436
04437
04438
          /* Read data... */
04439
04440
          if (in != NULL)
            while (fgets(line, LEN, in))
  if (sscanf(line, "%s %s %s", rvarname, dummy, rval) == 3)
  if (strcasecmp(rvarname, fullname1) == 0 ||
    strcasecmp(rvarname, fullname2) == 0) {
04441
04442
04443
04444
04445
                     contain = 1;
04446
                     break;
04447
          for (i = 1; i < argc - 1; i++)
04448
           if (strcasecmp(argv[i], fullname1) == 0 ||
    strcasecmp(argv[i], fullname2) == 0) {
04449
04450
04451
                sprintf(rval, "%s", argv[i + 1]);
```

```
04452
           contain = 1;
04453
           break;
04454
04455
       /* Close file... */
04456
       if (in != NULL)
04457
         fclose(in);
04459
04460
       /* Check for missing variables... */
04461
        if (!contain) {
        if (strlen(defvalue) > 0)
04462
           sprintf(rval, "%s", defvalue);
04463
04464
04465
           sprintf(msg, "Missing variable %s!\n", fullname1);
04466
            ERRMSG(msg);
04467
04468
04469
04470
       /* Write info... */
04471
       printf("%s = %s\n", fullname1, rval);
04472
04473
        /* Return values... */
       if (value != NULL)
    sprintf(value, "%s", rval);
04474
04475
04476
       return atof(rval);
04477 }
```

5.15.2.41 void tangent\_point (  $los_t * los_t$ , double \*  $tpz_t$ , double \*  $tplon_t$ , double \*  $tplon_t$ )

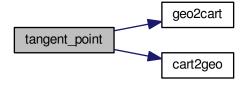
Find tangent point of a given LOS.

Definition at line 4481 of file jurassic.c.

```
04485
04486
04487
          double a, b, c, dummy, v[3], v0[3], v2[3], x, x1, x2, yy0, yy1, yy2;
04488
04489
          size_t i, ip;
04490
04491
          /* Find minimum altitude... */
04492
          ip = gsl_stats_min_index(los->z, 1, (size_t) los->np);
04493
04494
          /* Nadir or zenith... */
         if (ip <= 0 || ip >= (size_t) los->np - 1) {
  *tpz = los->z[los->np - 1];
  *tplon = los->lon[los->np - 1];
04495
04496
04497
04498
            *tplat = los->lat[los->np - 1];
04499
04500
04501
          /* Limb... */
04502
          else {
04503
             /* Determine interpolating polynomial y=a*x^2+b*x+c... */
04505
            yy0 = los -> z[ip - 1];
04506
            yy1 = los -> z[ip];
            yy2 = los -> z[ip + 1];
04507
            yy2 - 103-72[1p + 1],
x1 = sqrt(gsl_pow_2(los->ds[ip]) - gsl_pow_2(yy1 - yy0));
x2 = x1 + sqrt(gsl_pow_2(los->ds[ip + 1]) - gsl_pow_2(yy2 - yy1));
a = 1 / (x1 - x2) * (-(yy0 - yy1) / x1 + (yy0 - yy2) / x2);
b = -(yy0 - yy1) / x1 - a * x1;
04508
04509
04510
04511
04512
04513
04514
             /* Get tangent point location... */
            x = -b / (2 * a);

*tpz = a * x * x + b * x + c;
04515
04517
             geo2cart(los->z[ip - 1], los->lon[ip - 1], los->lat[ip - 1], v0);
04518
             geo2cart(los->z[ip + 1], los->lon[ip + 1], los->lat[ip + 1], v2);
            for (i = 0; i < 3; i++)
v[i] = LIN(0.0, v0[i], x2, v2[i], x);
04519
04520
04521
            cart2geo(v, &dummy, tplon, tplat);
04522
04523 }
```

Here is the call graph for this function:



5.15.2.42 void time2jsec (int year, int mon, int day, int hour, int min, int sec, double remain, double \* jsec )

Convert date to seconds.

Definition at line 4527 of file jurassic.c.

```
04535
04537
        struct tm t0, t1;
04538
04539
        t0.tm\_year = 100;
        t0.tm_mon = 0;
04540
        t0.tm_mday = 1;
04541
04542
        t0.tm\_hour = 0;
        t0.tm_min = 0;
t0.tm_sec = 0;
04543
04544
04545
        t1.tm_year = year - 1900;
t1.tm_mon = mon - 1;
04546
04547
04548
        t1.tm_mday = day;
        t1.tm_hour = hour;
04549
04550
        t1.tm_min = min;
04551
        t1.tm_sec = sec;
04552
        *jsec = (double) timegm(&t1) - (double) timegm(&t0) + remain;
04553
04554 }
```

5.15.2.43 void timer ( const char \* name, const char \* file, const char \* func, int line, int mode )

Measure wall-clock time.

Definition at line 4558 of file jurassic.c.

```
04563
04564
        static double dt_w, w0[10];
04565
04566
04567
        static int 10[10], nt;
04568
04569
        struct timeval tim;
04570
04571
         /* Start new timer... */
04572
        if (mode == 1) {
          gettimeofday(&tim, NULL);
          w0[nt] = (double) tim.tv_sec + (double) tim.tv_usec / 1e6;
10[nt] = line;
04574
04575
           if ((++nt) >= 10)
    ERRMSG("Too many timers!");
04576
04577
04578
04579
04580
        /* Write elapsed time... */
```

```
04581
       else {
04582
04583
          /\star Check timer index... \star/
          if (nt - 1 < 0)
04584
           ERRMSG("Coding error!");
04585
04586
04587
          /* Get time differences... */
04588
          gettimeofday(&tim, NULL);
04589
          dt_w = (double) tim.tv_sec + (double) tim.tv_usec / 1e6 - w0[nt - 1];
04590
04591
          /* Write elapsed time... */
         printf("Timer '%s' (%s, %s, 1%d-%d): %.3f sec\n",
04592
                 name, file, func, 10[nt - 1], line, dt_w);
04593
04594
04595
04596
        /* Stop timer... */
        if (mode == 3)
04597
04598
          nt--;
04599 }
```

5.15.2.44 void write\_atm ( const char \* dirname, const char \* filename, ctl\_t \* ctl, atm\_t \* atm\_)

Write atmospheric data.

Definition at line 4603 of file jurassic.c.

```
04607
                             {
04608
          FILE *out;
04609
04610
04611
          char file[LEN];
04612
04613
          int iq, ip, iw, n = 6;
04614
04615
           /* Set filename... */
04616
          if (dirname != NULL)
04617
             sprintf(file, "%s/%s", dirname, filename);
04618
          else
04619
             sprintf(file, "%s", filename);
04620
04621
          /* Write info... */
04622
          printf("Write atmospheric data: %s\n", file);
04624
           /* Create file... */
          if (!(out = fopen(file, "w")))
04625
             ERRMSG("Cannot create file!");
04626
04627
04628
           /* Write header... */
04629
04630
                      "# $1 = time (seconds since 2000-01-01T00:00Z) \n"
                      "# $2 = altitude [km] \n"
04631
                      "# $3 = longitude [deg]\n"
04632
                      "# $4 = latitude [deg]\n"
04633
                      "# $5 = pressure [hPa]\n" "# $6 = temperature [K]\n");
04634
          for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, "# $%d = %s volume mixing ratio\n", ++n, ctl->emitter[ig]);
04636
          for (iw = 0; iw < ctl->nw; iw++)
  fprintf(out, "# $%d = window %d: extinction [1/km]\n", ++n, iw);
04637
04638
04639
04640
           /* Write data... */
          for (ip = 0; ip < atm->np; ip++) {
04641
            for (ip = 0; ip < atm->np; ip++) {
    if (ip == 0 || atm->lat[ip] != atm->lat[ip - 1]
        || atm->lon[ip] != atm->lon[ip - 1])
    fprintf(out, "\n");
    fprintf(out, "%.2f %g %g %g %g", atm->time[ip], atm->z[ip],
        atm->lon[ip], atm->lat[ip], atm->p[ip], atm->t[ip]);
    for (iz = 0. iz < ctl->ng. ig++)
04643
04644
04645
04646
             for (ig = 0; ig < ctl->ng; ig++)
  fprintf(out, " %g", atm->q[ig][ip]);
04647
04648
             for (iw = 0; iw < ctl->nw; iw++)
  fprintf(out, " %g", atm->k[iw][ip]);
fprintf(out, "\n");
04649
04650
04651
04652
04653
           /* Close file... */
04654
04655
          fclose(out);
04656 }
```

5.15.2.45 void write\_matrix ( const char \* dirname, const char \* filename, ctl\_t \* ctl, gsl\_matrix \* matrix, atm\_t \* atm, obs\_t \* obs, const char \* rowspace, const char \* colspace, const char \* sort )

Write matrix.

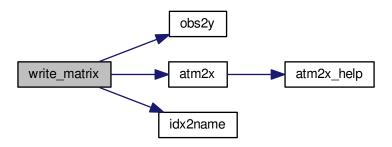
Definition at line 4660 of file jurassic.c.

```
04669
04670
04671
        FILE *out;
04672
04673
       char file[LEN], quantity[LEN];
04674
        int *cida, *ciqa, *cipa, *cira, *rida, *riqa, *ripa, *rira;
04676
04677
       size_t i, j, nc, nr;
04678
04679
        /* Check output flag... */
04680
        if (!ctl->write_matrix)
04681
         return;
04682
04683
        /* Allocate... */
04684
       ALLOC(cida, int, M);
04685
        ALLOC(ciqa, int,
04686
             N);
04687
        ALLOC(cipa, int,
04688
             N);
04689
        ALLOC(cira, int,
04690
             M);
        ALLOC(rida, int,
04691
04692
             M);
04693
        ALLOC(riqa, int,
04694
              N);
04695
        ALLOC(ripa, int,
04696
             N);
        ALLOC(rira, int,
04697
04698
             M);
04699
04700
        /* Set filename... */
04701
        if (dirname != NULL)
04702
         sprintf(file, "%s/%s", dirname, filename);
04703
        else
          sprintf(file, "%s", filename);
04704
04705
04706
        /* Write info... */
04707
       printf("Write matrix: %s\n", file);
04708
04709
        /* Create file... */
       if (!(out = fopen(file, "w")))
04710
04711
         ERRMSG("Cannot create file!");
04712
04713
        /* Write header (row space)... */
04714
       if (rowspace[0] == 'y') {
04715
04716
          fprintf(out.
04717
                   "# $1 = Row: index (measurement space) \n"
04718
                  "# $2 = Row: channel wavenumber [cm^-1]\n"
04719
                  "# $3 = \text{Row: time (seconds since } 2000-01-01T00:00Z) \n"
04720
                  "# $4 = Row: view point altitude [km]\n"
04721
                  "# $5 = Row: view point longitude [deg] \n"
                  "# $6 = Row: view point latitude [deg]\n");
04722
04723
04724
          /* Get number of rows... */
04725
          nr = obs2y(ctl, obs, NULL, rida, rira);
04726
04727
       } else {
04728
04729
          fprintf(out,
04730
                   "# $1 = Row: index (state space) \n"
04731
                  "# $2 = Row: name of quantity\n"
04732
                  "# $3 = Row: time (seconds since 2000-01-01T00:00Z)\n"
04733
                  "# $4 = Row: altitude [km]\n"
                  "# $5 = Row: longitude [deg]\n" "# $6 = Row: latitude [deg]\n");
04734
04735
04736
          /* Get number of rows... */
         nr = atm2x(ctl, atm, NULL, riqa, ripa);
04737
04738
04739
04740
        /\star Write header (column space)... \star/
04741
        if (colspace[0] == 'y') {
04742
04743
          fprintf(out,
04744
                  "# $7 = Col: index (measurement space) \n"
```

```
04745
                   "# $8 = Col: channel wavenumber [cm^-1]\n"
04746
                   "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04747
                    "# $10 = Col: view point altitude [km] \n"
04748
                   "# $11 = Col: view point longitude [deg]\n"
                   "# $12 = Col: view point latitude [deg]\n");
04749
04750
04751
           /* Get number of columns... */
04752
          nc = obs2y(ctl, obs, NULL, cida, cira);
04753
04754
        } else {
04755
04756
          fprintf(out,
                    "# $7 = Col: index (state space)\n"
04757
04758
                   "# $8 = Col: name of quantity\n"
04759
                   "# $9 = Col: time (seconds since 2000-01-01T00:00Z)\n"
04760
                   "# $10 = Col: altitude [km] \n"
                   "# \$11 = Col: longitude [deg]\n" "# \$12 = Col: latitude [deg]\n");
04761
04762
04763
           /\star Get number of columns... \star/
04764
          nc = atm2x(ctl, atm, NULL, ciqa, cipa);
04765
04766
        /* Write header entry... */ fprintf(out, "# $13 = Matrix element\n\n");
04767
04768
04769
04770
        /* Write matrix data... */
04771
        i = j = 0;
04772
        while (i < nr && j < nc) {</pre>
04773
04774
           /* Write info about the row... */
          04775
04777
04778
                     obs->time[rira[i]], obs->vpz[rira[i]],
04779
                     obs->vplon[rira[i]], obs->vplat[rira[i]]);
04780
           else {
04781
             idx2name(ctl, riqa[i], quantity);
fprintf(out, "%d %s %.2f %g %g %g", (int) i, quantity,
04782
04783
                     atm->time[ripa[i]], atm->z[ripa[i]],
04784
                     atm->lon[ripa[i]], atm->lat[ripa[i]]);
04785
04786
          /* Write info about the column... */
if (colspace[0] == 'y')
  fprintf(out, " %d %g %.2f %g %g %g",
04787
04788
04789
04790
                     (int) j, ctl->nu[cida[j]],
04791
                     obs->time[cira[j]], obs->vpz[cira[j]],
04792
                     obs->vplon[cira[j]], obs->vplat[cira[j]]);
04793
          else {
            idx2name(ctl, ciqa[j], quantity);
fprintf(out, " %d %s %.2f %g %g %g", (int) j, quantity,
04794
04795
04796
                     atm->time[cipa[j]], atm->z[cipa[j]]
04797
                     atm->lon[cipa[j]], atm->lat[cipa[j]]);
04798
          }
04799
          /* Write matrix entry... */ fprintf(out, " g^n, gsl_matrix_get(matrix, i, j));
04800
04801
04802
04803
           /* Set matrix indices... */
04804
           if (sort[0] == 'r') {
04805
             j++;
             if (j >= nc) {
04806
04807
              j = 0;
04808
04809
               fprintf(out, "\n");
04810
04811
          } else {
04812
             i++;
04813
             if (i >= nr) {
              i = 0;
04814
04815
               j++;
04816
               fprintf(out, "\n");
04817
            }
04818
          }
04819
        }
04820
04821
         /* Close file... */
04822
        fclose(out);
04823
04824
        /* Free... */
04825
        free(cida);
04826
        free (ciqa);
04827
        free(cipa);
04828
        free (cira);
04829
        free(rida);
04830
        free (riga);
04831
        free (ripa):
```

```
04832 free(rira);
04833 }
```

Here is the call graph for this function:



5.15.2.46 void write\_obs ( const char \* dirname, const char \* filename, ctl t \* ctl, obs t \* obs)

Write observation data.

Definition at line 4837 of file jurassic.c.

```
04841
04842
04843
        FILE *out;
04844
04845
        char file[LEN];
04846
04847
        int id, ir, n = 10;
04848
04849
        /* Set filename... */
04850
        if (dirname != NULL)
04851
          sprintf(file, "%s/%s", dirname, filename);
04852
          sprintf(file, "%s", filename);
04853
04854
04855
        /* Write info... */
04856
        printf("Write observation data: %s\n", file);
04857
04858
        /* Create file... */
        if (!(out = fopen(file, "w")))
04859
04860
          ERRMSG("Cannot create file!");
04861
04862
        /* Write header... */
04863
        fprintf(out,
04864
                 "# $1 = time (seconds since 2000-01-01T00:00Z) \n"
                 "# $2 = observer altitude [km] \n"
04865
                 "# $3 = observer longitude [deg]\n"
04866
                 "# $4 = observer latitude [deg]\n'
04867
                 "# $5 = view point altitude [km]\n"
04868
                 "# $6 = view point longitude [deg]\n"
04869
04870
                 "# $7 = view point latitude [deg]\n"
                 "# $8 = tangent point altitude [km]\n" "# $9 = tangent point longitude [deg]\n"
04871
04872
                 "# $10 = tangent point latitude [deg]\n");
04873
04874
        for (id = 0; id < ctl->nd; id++)
04875
         fprintf(out, "# $%d = channel %g: radiance [W/(m^2 sr cm^-1)]\n", ++n, ctl->nu[id]);
04876
        for (id = 0; id < ctl->nd; id+)
  fprintf(out, "# $%d = channel %g: transmittance\n", ++n, ctl->nu[id]);
04877
04878
04879
04880
        /* Write data... */
04881
        for (ir = 0; ir < obs->nr; ir++) {
04882
          if (ir == 0 || obs->time[ir] != obs->time[ir - 1])
```

```
fprintf(out, "\n");
fprintf(out, "%.2f %g %g %g %g %g %g %g %g %g %g", obs->time[ir],
04883
04884
04885
                         obs->obsz[ir], obs->obslon[ir], obs->obslat[ir],
                        obs->vpz[ir], obs->vplon[ir], obs->vplat[ir],
04886
             obs->tpz[ir], obs->tplon[ir], obs->tplat[ir]);
for (id = 0; id < ctl->nd; id++)
  fprintf(out, " %g", obs->rad[id][ir]);
04887
04888
04889
             for (id = 0; id < ctl->nd; id++)
  fprintf(out, " %g", obs->tau[id][ir]);
fprintf(out, "\n");
04890
04891
04892
04893
04894
04895
           /* Close file... */
04896
          fclose(out);
04897 }
```

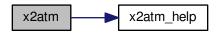
5.15.2.47 void x2atm ( ctl\_t \* ctl, gsl\_vector \* x, atm\_t \* atm )

Decompose parameter vector or state vector.

Definition at line 4901 of file jurassic.c.

```
04904
04905
04906
       int iq, iw;
04907
       size_t n = 0;
04908
04909
04910
       /* Set pressure... */
       x2atm_help(atm, ctl->retp_zmin, ctl->retp_zmax, atm->
04911
     p, x, &n);
04912
04913
        /* Set temperature... */
04914
       x2atm_help(atm, ctl->rett_zmin, ctl->rett_zmax, atm->
     t, x, &n);
04915
04916
       /* Set volume mixing ratio... */
       for (ig = 0; ig < ctl->ng; ig++)
04917
04918
         x2atm_help(atm, ctl->retq_zmin[ig], ctl->retq_zmax[ig],
04919
                     atm->q[ig], x, &n);
04920
04921
       /* Set extinction... */
04922
       for (iw = 0; iw < ctl->nw; iw++)
         x2atm_help(atm, ctl->retk_zmin[iw], ctl->retk_zmax[iw],
04923
04924
                     atm->k[iw], x, &n);
04925 }
```

Here is the call graph for this function:



5.15.2.48 void x2atm\_help ( atm\_t \* atm, double zmin, double zmax, double \* value, gsl\_vector \* x, size\_t \* n )

Extract elements from state vector.

Definition at line 4929 of file jurassic.c.

```
04935
04936
04937
        int ip;
04938
04939
        /* Extract state vector elements... */
04940
        for (ip = 0; ip < atm->np; ip++)
         if (atm->z[ip] >= zmin && atm->z[ip] <= zmax) {</pre>
04941
04942
            value[ip] = gsl_vector_get(x, *n);
04943
            (*n)++;
04944
04945 }
```

```
5.15.2.49 void y2obs ( ctl_t * ctl, gsl_vector * y, obs_t * obs )
```

Decompose measurement vector.

Definition at line 4949 of file jurassic.c.

```
04952
04953
04954
       int id, ir:
04955
04956
       size_t m = 0;
04957
04958
        /* Decompose measurement vector... */
04959
       for (ir = 0; ir < obs->nr; ir++)
        for (id = 0; id < ctl->nd; id++)
04960
04961
           if (gsl_finite(obs->rad[id][ir])) {
04962
             obs->rad[id][ir] = gsl_vector_get(y, m);
04963
             m++;
04964
04965 }
```

## 5.16 jurassic.h

```
00001 /*
00002
         This file is part of JURASSIC.
00003
00004
         JURASSIC is free software: you can redistribute it and/or modify
00005
         it under the terms of the GNU General Public License as published by
00006
         the Free Software Foundation, either version 3 of the License, or
00007
         (at your option) any later version.
80000
        JURASSIC is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00009
00010
00011
00012
        GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
         Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00034 #include <math.h>
00035 #include <stdio.h>
00036 #include <stdlib.h>
00037 #include <string.h>
00038 #include <time.h>
00039 #include <sys/time.h>
00040 #include <gsl/gsl_math.h>
00041 #include <gsl/gsl_blas.h>
00042 #include <gsl/gsl_const_mksa.h>
00043 #include <gsl/gsl_const_num.h>
00044 #include <gsl/gsl_linalg.h>
00045 #include <gsl/gsl_statistics.h>
00046
00047 /* -----
00048
         Macros...
00049
00050
00052 #define ALLOC(ptr, type, n)
00053 if((ptr=malloc((size_t)(n)*sizeof(type)))==NULL)
00054
          ERRMSG("Out of memory!");
00055
00057 #define DIST(a, b) sqrt(DIST2(a, b))
00058
00060 #define DIST2(a, b)
00061
         ((a[0]-b[0])*(a[0]-b[0])+(a[1]-b[1])*(a[1]-b[1])+(a[2]-b[2])*(a[2]-b[2]))
00062
00064 #define DOTP(a, b) (a[0]*b[0]+a[1]*b[1]+a[2]*b[2])
00065
00067 #define ERRMSG(msg) {
         printf("\nError (%s, %s, 1%d): %s\n\n",
00068
00069
                     _FILE__, __func__, __LINE__, msg);
00070
           exit(EXIT_FAILURE);
00071
00072
00074 #define EXP(x0, y0, x1, y1, x)
00075 (((y0)>0 && (y1)>0)
          ? ((y0) * exp(log((y1)/(y0))/((x1)-(x0))*((x)-(x0)))
```

5.16 jurassic.h 235

```
00077
       : LIN(x0, y0, x1, y1, x))
00078
00080 #define LIN(x0, y0, x1, y1, x)
00081 ((y0) + ((y1) - (y0)) / ((x1) - (x0)) * ((x) - (x0)))
00082
00084 #define NORM(a) sgrt(DOTP(a, a))
00085
00087 #define PRINT(format, var)
00088 printf("Print (%s, %s, 1%d): %s= "format"n",
00089
              __FILE__, __func__, __LINE__, #var, var);
00090
00092 #define TIMER(name, mode)
       {timer(name, __FILE__, __func__, __LINE__, mode);}
00093
00094
00100
00101
00102 /* -----
       Constants...
00103
00104
00105
00107 #define C1 1.19104259e-8
00108
00110 #define C2 1.43877506
00111
00113 #define TMIN 100.
00114
00116 #define TMAX 400.
00117
00119 #define G0 9.80665
00120
00122 #define P0 1013.25
00123
00125 #define T0 273.15
00128 #define RE 6367.421
00129
00131 #define ME 5.976e24
00132
00133 /* -
00134
       Dimensions...
00135
00136
00138 #define ND 50
00139
00141 #define NG 20
00142
00144 #define NP 1000
00145
00147 #define NR 1000
00148
00150 #define NW 5
00151
00153 #define LEN 5000
00154
00156 #define M (NR*ND)
00157
00159 #define N (NO*NP)
00160
00162 #define NQ (2+NG+NW)
00163
00165 #define NLOS 1000
00166
00168 #define NSHAPE 10000
00169
00171 #define NFOV 5
00172
00174 #define TBLNP 41
00175
00177 #define TBLNT 30
00178
00180 #define TBLNU 320
00181
00183 #define TBLNS 1200
00184
00185 /* -----
00186
       Quantity indices...
00187
00188
00190 #define IDXP 0
00191
00193 #define IDXT 1
00194
00196 #define IDXQ(iq) (2+iq)
```

```
00197
00199 #define IDXK(iw) (2+ctl->ng+iw)
00200
00201 /* -----
00202
        Structs...
00203
00204
00206 typedef struct {
00207
00209
       int np;
00210
00212
       double time[NP];
00213
00215
       double z[NP];
00216
00218
00219
       double lon[NP];
00221
       double lat[NP];
00222
00224
       double p[NP];
00225
00227
       double t[NP];
00228
00230
       double q[NG][NP];
00231
00233
       double k[NW][NP];
00234
00235 } atm_t;
00236
00238 typedef struct {
00239
00241
       int ng;
00242
00244
       char emitter[NG][LEN];
00245
00247
       int nd;
00248
       int nw;
00251
00253
       double nu[ND];
00254
00256
       int window[ND];
00257
00259
       char tblbase[LEN];
00260
00262
       double hydz;
00263
00265
       int ctm_co2;
00266
00268
       int ctm h2o:
00269
00271
       int ctm_n2;
00272
00274
00275
       int ctm_o2;
00277
       int refrac;
00278
00280
       double rayds;
00281
00283
       double raydz;
00284
00286
       char fov[LEN];
00287
00289
       double retp_zmin;
00290
00292
       double retp_zmax;
00293
00295
       double rett_zmin;
00296
       double rett_zmax;
00299
00301
       double retq_zmin[NG];
00302
00304
       double retq_zmax[NG];
00305
       double retk_zmin[NW];
00308
00310
       double retk_zmax[NW];
00311
00313
       int write bbt;
00314
00316
       int write_matrix;
00317
00318 } ctl_t;
00319
00321 typedef struct {
00322
```

5.16 jurassic.h 237

```
00324
        int np;
00325
00327
        double z[NLOS];
00328
        double lon[NLOS];
00331
00333
        double lat[NLOS];
00334
00336
        double p[NLOS];
00337
        double t[NLOS];
00339
00340
00342
        double q[NG][NLOS];
00343
00345
        double k[NW][NLOS];
00346
00348
        double tsurf;
00349
00351
       double ds[NLOS];
00352
00354
        double u[NG][NLOS];
00355
00356 } los_t;
00357
00359 typedef struct {
00360
00362
        int nr;
00363
00365
        double time[NR];
00366
00368
       double obsz[NR];
00369
00371
       double obslon[NR];
00372
00374
        double obslat[NR];
00375
00377
        double vpz[NR];
00378
00380
        double vplon[NR];
00381
00383
        double vplat[NR];
00384
00386
        double tpz[NR];
00387
00389
        double tplon[NR];
00390
00392
        double tplat[NR];
00393
00395
        double tau[ND][NR];
00396
00398
        double rad[ND][NR];
00399
00400 } obs_t;
00401
00403 typedef struct {
00404
00406
        int np[NG][ND];
00407
00409
        int nt[NG][ND][TBLNP];
00410
00412
        int nu[NG][ND][TBLNP][TBLNT];
00413
00415
        double p[NG][ND][TBLNP];
00416
00418
        double t[NG][ND][TBLNP][TBLNT];
00419
00421
        float u[NG][ND][TBLNP][TBLNT][TBLNU];
00422
00424
        float eps[NG][ND][TBLNP][TBLNT][TBLNU];
00425
00427
        double st[TBLNS];
00428
00430
        double sr[ND][TBLNS];
00431
00432 } tbl_t;
00434 /* --
00435
         Functions...
00436
00437
00439 size_t atm2x(
        ctl_t * ctl,
atm_t * atm,
00440
00441
00442
        gsl\_vector * x,
00443
        int *iqa,
        int *ipa);
00444
00445
```

```
00447 void atm2x_help(
       atm_t * atm,
double zmin,
00448
00449
00450
        double zmax,
00451
        double *value,
        int val_iqa,
00452
00453
        gsl_vector * x,
00454
         int *iqa,
00455
        int *ipa,
00456
        size_t * n);
00457
00459 double brightness(
00460
        double rad,
00461
        double nu);
00462
00464 void cart2geo(
00465
        double *x,
00466
        double *z,
double *lon,
00467
00468
        double *lat);
00469
00471 void climatology(
00472
       ctl_t * ctl,
atm_t * atm_mean);
00473
00474
00476 double ctmco2(
00477
        double nu,
00478
        double p,
00479
        double t,
00480
        double u);
00481
00483 double ctmh2o(
00484
        double nu,
00485
        double p,
00486
        double t,
00487
        double q,
00488
        double u);
00489
00491 double ctmn2(
00492
        double nu,
00493
        double p,
00494
        double t);
00495
00497 double ctmo2(
00498
        double nu,
00499
        double p,
00500
        double t);
00501
00503 void copy_atm(
        ctl_t * ctl,
atm_t * atm_dest,
00504
00505
00506
        atm_t * atm_src,
00507
        int init);
00508
00510 void copy_obs(
        ctl_t * ctl,
obs_t * obs_dest,
obs_t * obs_src,
00511
00512
00513
00514
        int init);
00515
00517 int find_emitter(
00518 ctl_t * ctl,
00519 const char *emitter);
00520
00522 void formod(
       ctl_t * ctl,
atm_t * atm,
00523
00524
        obs_t * obs);
00525
00526
00528 void formod_continua(
        ctl_t * ctl,
los_t * los,
00529
00530
00531
        int ip,
00532
        double *beta);
00533
00535 void formod_fov(
00536
        ctl_t * ctl,
        obs_t * obs);
00537
00538
00540 void formed pencil(
        ctl_t * ctl,
atm_t * atm,
00541
00542
        obs_t * obs,
00543
00544
        int ir);
00545
00547 void formod_srcfunc(
00548
       ctl_t * ctl,
```

5.16 jurassic.h 239

```
tbl_t * tbl,
00550
        double t,
00551
        double *src);
00552
00554 void geo2cart(
00555
        double z.
        double lon,
00557
        double lat,
00558
        double *x);
00559
00561 double gravity(
        double z,
00562
00563
        double lat);
00564
00566 void hydrostatic(
       ctl_t * ctl,
atm_t * atm);
00567
00568
00569
00571 void idx2name(
00572
        ctl_t * ctl,
00573
        int idx,
00574
        char *quantity);
00575
00577 void init_tbl(
00578 ctl_t * ctl,
00579 tbl_t * tbl);
00580
00582 void intpol_atm(
        ctl_t * ctl,
atm_t * atm,
00583
00584
00585
        double z,
00586
        double *p,
00587
        double *t,
00588
        double *q,
00589
        double *k);
00590
00592 void intpol_tbl(
        ctl_t * ctl,
tbl_t * tbl,
00594
00595
        los_t * los,
00596
        int ip,
00597
        double tau_path[NG][ND],
00598
        double tau_seg[ND]);
00599
00601 double intpol_tbl_eps(
00602
        tbl_t * tbl,
00603
        int ig,
00604
        int id,
00605
        int ip,
00606
        int it,
00607
        double u);
00608
00610 double intpol_tbl_u(
00611
        tbl_t * tbl,
00612
        int iq,
00613
        int id,
00614
        int ip,
00615
        int it,
00616
        double eps);
00617
00619 void jsec2time(
00620
        double isec,
00621
        int *year,
00622
        int *mon,
        int *day,
00623
00624
        int *hour,
00625
        int *min,
int *sec,
00626
00627
        double *remain);
00628
00630 void kernel(
        ctl_t * ctl,
atm_t * atm,
obs_t * obs,
00631
00632
00633
00634
        gsl_matrix * k);
00635
00637 int locate(
00638 double *xx,
00639
        int n,
        double x);
00640
00641
00643 int locate_tbl(
00644
        float *xx,
00645
        int n,
00646
        double x);
00647
00649 size_t obs2y(
```

```
00650
        ctl_t * ctl,
obs_t * obs,
00651
00652
        gsl_vector * y,
00653
        int *ida,
00654
        int *ira);
00655
00657 double planck(
00658
        double t,
00659
        double nu);
00660
00662 void raytrace(
        ctl_t * ctl,
atm_t * atm,
00663
00664
        obs_t * obs,
los_t * los,
00665
00666
00667
        int ir);
00668
00670 void read_atm(
       const char *dirname, const char *filename,
00671
00672
        ctl_t * ctl,
atm_t * atm);
00673
00674
00675
00677 void read_ctl(
00678
        int argc,
00679
        char *argv[],
00680
        ctl_t * ctl);
00681
00683 void read_matrix(
00684
       const char *dirname,
const char *filename,
00685
00686
        gsl_matrix * matrix);
00687
00689 void read_obs(
       const char *dirname,
const char *filename,
00690
00691
        ctl_t * ctl,
obs_t * obs);
00692
00693
00694
00696 void read_shape(
00697
        const char *filename,
        double *x, double *y,
00698
00699
00700
        int *n);
00701
00703 double refractivity(
00704 double p,
00705
        double t);
00706
00708 double scan_ctl(
00709
        int argc,
00710
        char *argv[],
00711
        const char *varname,
        int arridx,
const char *defvalue,
00712
00713
00714
        char *value);
00715
00717 void tangent_point(
double *tplat);
00721
00722
00724 void time2jsec(
00725
        int year,
00726
        int mon,
00727
        int day,
00728
        int hour,
00729
        int min,
00730
        int sec,
00731
        double remain,
00732
        double *jsec);
00733
00735 void timer(
00736
        const char *name,
00737
        const char *file,
00738
        const char *func,
00739
        int line,
00740
        int mode);
00741
00743 void write_atm(
        const char *dirname, const char *filename,
00744
00745
00746
        ctl_t * ctl,
00747
        atm_t * atm);
00748
00750 void write_matrix(
```

```
const char *dirname,
00752
         const char *filename,
00753
         ctl_t * ctl,
        gsl_matrix * matrix,
00754
00755
        atm_t * atm,
obs_t * obs,
00756
        const char *rowspace,
00758
        const char *colspace,
00759
        const char *sort);
00760
00762 void write_obs(
00763 const char *dirname,
00764 const char *filename,
        ctl_t * ctl,
obs_t * obs);
00765
00766
00767
00769 void x2atm(
00770 ctl_t * ctl,
00771 gsl_vector * x,
00772
        atm_t * atm);
00773
00775 void x2atm_help(
00776 atm_t * atm,
00777 double zmin,
00778
        double zmax,
00779
        double *value,
00780
        gsl_vector * x,
00781
00782
00784 void y2obs(
00785 ctl_t * ctl,
00786 gsl_vector * y,
00787 obs_t * obs);
```

### 5.17 kernel.c File Reference

Calculate kernel functions.

# **Functions**

• int main (int argc, char \*argv[])

### 5.17.1 Detailed Description

Calculate kernel functions.

Definition in file kernel.c.

### 5.17.2 Function Documentation

## 5.17.2.1 int main ( int argc, char \* argv[] )

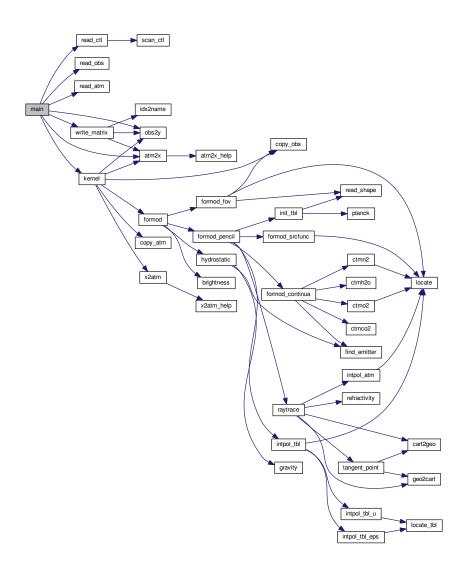
Definition at line 27 of file kernel.c.

```
00029 {
00030
00031 static atm_t atm;
00032 static ctl_t ctl;
00033 static obs_t obs;
00034
00035 gsl_matrix *k;
00036
00037 size_t m, n;
00038
00039 /* Check arguments... */
```

```
00040
        if (argc < 5)
00041
          ERRMSG("Give parameters: <ctl> <obs> <atm> <kernel>");
00042
00043
        /\star Read control parameters... \star/
00044
        read_ctl(argc, argv, &ctl);
00045
00046
        /* Set flags... */
00047
        ctl.write_matrix = 1;
00048
       /* Read observation geometry... */
read_obs(NULL, argv[2], &ctl, &obs);
00049
00050
00051
00052
        /* Read atmospheric data... */
00053
        read_atm(NULL, argv[3], &ctl, &atm);
00054
       /* Get sizes... */
n = atm2x(&ctl, &atm, NULL, NULL, NULL);
m = obs2y(&ctl, &obs, NULL, NULL, NULL);
00055
00056
00057
00058
00059
        /* Check sizes... */
00060
        if (n <= 0)
00061
          ERRMSG("No state vector elements!");
00062
        if (m \ll 0)
00063
         ERRMSG("No measurement vector elements!");
00064
00065
        /* Allocate... */
00066
        k = gsl_matrix_alloc(m, n);
00067
        /* Compute kernel matrix... */
00068
00069
        kernel(&ctl, &atm, &obs, k);
00070
00071
        /* Write matrix to file... */
00072
        write_matrix(NULL, argv[4], &ctl, k, &atm, &obs, "y", "x", "r");
00073
00074
        /* Free... */
00075
        gsl_matrix_free(k);
00076
00077
        return EXIT_SUCCESS;
00078 }
```

5.18 kernel.c 243

Here is the call graph for this function:



## 5.18 kernel.c

```
00001 /*
00002
         This file is part of JURASSIC.
00003
00004
         JURASSIC is free software: you can redistribute it and/or modify
00005
         it under the terms of the GNU General Public License as published by
00006
         the Free Software Foundation, either version 3 of the License, or
00007
         (at your option) any later version.
00008
00009
         JURASSIC is distributed in the hope that it will be useful,
         but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00010
00011
00012
         GNU General Public License for more details.
00013
00014
         You should have received a copy of the GNU General Public License \,
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028
        int argc,
```

```
00029
        char *argv[]) {
00030
00031
        static atm_t atm;
00032
        static ctl_t ctl;
00033
        static obs_t obs;
00034
        gsl_matrix *k;
00036
00037
        size_t m, n;
00038
00039
        /* Check arguments... */
00040
        if (argc < 5)
00041
          ERRMSG("Give parameters: <ctl> <obs> <atm> <kernel>");
00042
00043
        /\star Read control parameters... \star/
00044
        read_ctl(argc, argv, &ctl);
00045
00046
        /* Set flags... */
ctl.write_matrix = 1;
00047
00048
00049
         /* Read observation geometry... */
00050
        read_obs(NULL, argv[2], &ctl, &obs);
00051
00052
        /* Read atmospheric data... */
00053
        read_atm(NULL, argv[3], &ctl, &atm);
00054
00055
        /* Get sizes... */
        n = atm2x(&ctl, &atm, NULL, NULL, NULL);
m = obs2y(&ctl, &obs, NULL, NULL, NULL);
00056
00057
00058
00059
        /* Check sizes... */
00060
            (n <= 0)
00061
          ERRMSG("No state vector elements!");
00062
        if (m <= 0)
00063
          ERRMSG("No measurement vector elements!");
00064
00065
        /* Allocate... */
        k = gsl_matrix_alloc(m, n);
00066
00067
00068
        /* Compute kernel matrix... */
00069
        kernel(&ctl, &atm, &obs, k);
00070
        /* Write matrix to file... */
write_matrix(NULL, argv[4], &ctl, k, &atm, &obs, "y", "x", "r");
00071
00072
00073
00074
00075
        gsl_matrix_free(k);
00076
00077
        return EXIT_SUCCESS;
00078 }
```

### 5.19 limb.c File Reference

Create observation geometry for a limb sounder.

#### **Functions**

• int main (int argc, char \*argv[])

## 5.19.1 Detailed Description

Create observation geometry for a limb sounder.

Definition in file limb.c.

5.20 limb.c 245

### 5.19.2 Function Documentation

### 5.19.2.1 int main ( int *argc*, char \* *argv[*])

Definition at line 27 of file limb.c.

```
00029
00031
          static ctl_t ctl;
00032
          static obs_t obs;
00033
00034
          double dz, obsz, z, z0, z1;
00035
00036
          /* Check arguments... */
00037
          if (argc < 3)
00038
            ERRMSG("Give parameters: <ctl> <obs>");
00039
00040
          /* Read control parameters... */
          read_ctl(argc, argv, &ctl);

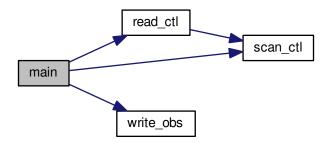
obsz = scan_ctl(argc, argv, "OBSZ", -1, "780", NULL);

z0 = scan_ctl(argc, argv, "Z0", -1, "6", NULL);

z1 = scan_ctl(argc, argv, "21", -1, "70", NULL);

dz = scan_ctl(argc, argv, "DZ", -1, "1.5", NULL);
00041
00042
00043
00044
00045
00046
00047
          /\star Create measurement geometry... \star/
          for (z = z0; z \le z1; z += dz) {
00048
00049
            obs.obsz[obs.nr] = obsz;
00050
             obs.vpz[obs.nr] = z;
00051
             obs.vplat[obs.nr] = 180 / M_PI * acos((RE + z) / (RE + obsz));
            if ((++obs.nr) >= NR)
   ERRMSG("Too many rays!");
00052
00053
00054
00055
00056
           /* Write observation data... */
00057
          write_obs(NULL, argv[2], &ctl, &obs);
00058
00059
          return EXIT_SUCCESS;
00060 }
```

Here is the call graph for this function:



#### 5.20 limb.c

```
00001 /*
00002 This file is part of JURASSIC.
00003
00004 JURASSIC is free software: you can redistribute it and/or modify
00005 it under the terms of the GNU General Public License as published by
00006 the Free Software Foundation, either version 3 of the License, or
00007 (at your option) any later version.
```

```
JURASSIC is distributed in the hope that it will be useful,
          but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00010
00011
00012
          GNU General Public License for more details.
00013
          You should have received a copy of the GNU General Public License
00014
00015
         along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
          Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028
         int argc,
00029
         char *argv[]) {
00030
         static ctl_t ctl;
static obs_t obs;
00031
00032
00033
00034
         double dz, obsz, z, z0, z1;
00035
00036
          /* Check arguments... */
00037
         if (argc < 3)
00038
            ERRMSG("Give parameters: <ctl> <obs>");
00039
00040
          /\star Read control parameters... \star/
         read_ctl(argc, argv, &ctl);

obsz = scan_ctl(argc, argv, "obsz", -1, "780", NULL);

z0 = scan_ctl(argc, argv, "z0", -1, "6", NULL);

z1 = scan_ctl(argc, argv, "z1", -1, "70", NULL);

dz = scan_ctl(argc, argv, "DZ", -1, "1.5", NULL);
00041
00042
00043
00044
00045
00046
00047
          /\star Create measurement geometry... \star/
00048
         for (z = z0; z \le z1; z += dz) {
00049
            obs.obsz[obs.nr] = obsz;
00050
            obs.vpz[obs.nr] = z;
00051
            obs.vplat[obs.nr] = 180 / M_PI * acos((RE + z) / (RE + obsz));
00052
            if ((++obs.nr) >= NR)
00053
               ERRMSG("Too many rays!");
00054
00055
00056
         /* Write observation data... */
         write_obs(NULL, argv[2], &ctl, &obs);
00057
00058
00059
          return EXIT_SUCCESS;
00060 }
```

## 5.21 nadir.c File Reference

Create observation geometry for a nadir sounder.

### **Functions**

• int main (int argc, char \*argv[])

### 5.21.1 Detailed Description

Create observation geometry for a nadir sounder.

Definition in file nadir.c.

5.22 nadir.c 247

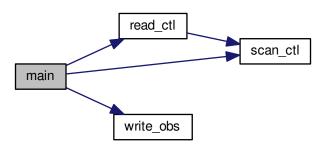
### 5.21.2 Function Documentation

### 5.21.2.1 int main ( int *argc*, char \* *argv[*])

Definition at line 27 of file nadir.c.

```
00029
00030
00031
           static ctl_t ctl;
00032
           static obs_t obs;
00033
00034
           double dlat, lat, lat0, lat1, obsz;
00035
00036
           /* Check arguments... */
00037
           if (argc < 3)
00038
             ERRMSG("Give parameters: <ctl> <obs>");
00039
00040
           /* Read control parameters... */
           read_ctl(argc, argv, &ctl);
obsz = scan_ctl(argc, argv, "OBSZ", -1, "700", NULL);
lat0 = scan_ctl(argc, argv, "LAT0", -1, "-8.01", NULL);
lat1 = scan_ctl(argc, argv, "LAT1", -1, "8.01", NULL);
dlat = scan_ctl(argc, argv, "DLAT", -1, "0.18", NULL);
00041
00042
00043
00044
00045
00046
00047
           /* Create measurement geometry... */
for (lat = lat0; lat <= lat1; lat += dlat) {</pre>
00048
00049
             obs.obsz[obs.nr] = obsz;
00050
              obs.vplat[obs.nr] = lat;
              if ((++obs.nr) >= NR)
   ERRMSG("Too many rays!");
00051
00052
00053
00054
00055
           /* Write observation data... */
00056
           write_obs(NULL, argv[2], &ctl, &obs);
00057
00058
           return EXIT_SUCCESS;
00059 }
```

Here is the call graph for this function:



# 5.22 nadir.c

```
00001 /*
00002 This file is part of JURASSIC.
00003
00004 JURASSIC is free software: you can redistribute it and/or modify
00005 it under the terms of the GNU General Public License as published by
00006 the Free Software Foundation, either version 3 of the License, or
00007 (at your option) any later version.
00008
00009 JURASSIC is distributed in the hope that it will be useful,
```

```
but WITHOUT ANY WARRANTY; without even the implied warranty of
00011
         MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00012
         GNU General Public License for more details.
00013
         You should have received a copy of the GNU General Public License
00014
00015
         along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
         Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028 int argc,
00029
        char *argv[]) {
00030
        static ctl_t ctl;
00031
00032
        static obs_t obs;
00033
00034
         double dlat, lat, lat0, lat1, obsz;
00035
00036
         /* Check arguments... */
         if (argc < 3)
00037
00038
           ERRMSG("Give parameters: <ctl> <obs>");
00039
00040
        /* Read control parameters... */
00041
         read_ctl(argc, argv, &ctl);
        obsz = scan_ctl(argc, argv, "OBSZ", -1, "700", NULL);
lat0 = scan_ctl(argc, argv, "LAT0", -1, "-8.01", NULL);
lat1 = scan_ctl(argc, argv, "LAT1", -1, "8.01", NULL);
dlat = scan_ctl(argc, argv, "DLAT", -1, "0.18", NULL);
00042
00043
00044
00045
00046
00047
         /* Create measurement geometry... */
00048
         for (lat = lat0; lat <= lat1; lat += dlat) {</pre>
         obs.obsz[obs.nr] = obsz;
00049
           obs.vplat[obs.nr] = lat;
00050
00051
          if ((++obs.nr) >= NR)
              ERRMSG("Too many rays!");
00052
00053
00054
00055
        /* Write observation data... */
        write_obs(NULL, argv[2], &ctl, &obs);
00056
00057
00058
        return EXIT_SUCCESS;
00059 }
```

## 5.23 planck.c File Reference

Convert brightness temperature to radiance.

### **Functions**

• int main (int argc, char \*argv[])

### 5.23.1 Detailed Description

Convert brightness temperature to radiance.

Definition in file planck.c.

### 5.23.2 Function Documentation

5.23.2.1 int main ( int argc, char \* argv[])

Definition at line 27 of file planck.c.

5.24 planck.c 249

```
00029
00030
00031
        double nu, t;
00032
00033
        /* Check arguments... */
00034
        if (argc < 3)
          ERRMSG("Give parameters: <t> <nu>");
00036
00037
        /* Read arguments... */
       t = atof(argv[1]);
00038
00039
       nu = atof(argv[2]);
00040
00041
        /* Compute Planck function... */
00042
        printf("%.10g\n", planck(t, nu));
00043
00044
        return EXIT_SUCCESS;
00045 }
```

Here is the call graph for this function:



# 5.24 planck.c

```
00001 /*
        This file is part of JURASSIC.
00003
00004
        {\tt JURASSIC} is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
00005
00006
00007
        (at your option) any later version.
00008
00009
        JURASSIC is distributed in the hope that it will be useful,
00010
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00011
00012
        GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028
        int argc,
00029
        char *argv[]) {
00030
00031
        double nu, t;
00032
00033
        /* Check arguments... */
00034
          ERRMSG("Give parameters: <t> <nu>");
00035
00036
00037
        /* Read arguments... */
00038
        t = atof(argv[1]);
00039
        nu = atof(argv[2]);
00040
00041
         /* Compute Planck function... */
00042
        printf("%.10g\n", planck(t, nu));
00043
00044
        return EXIT_SUCCESS;
00045 }
```

# 5.25 raytrace.c File Reference

Determine atmospheric ray paths.

#### **Functions**

• int main (int argc, char \*argv[])

## 5.25.1 Detailed Description

Determine atmospheric ray paths.

Definition in file raytrace.c.

#### 5.25.2 Function Documentation

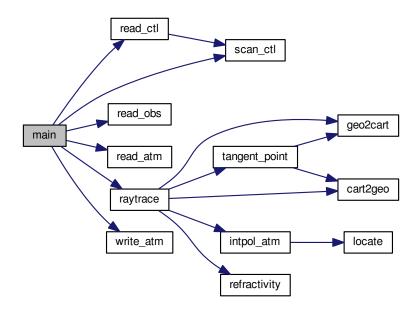
## 5.25.2.1 int main (int argc, char \* argv[])

Definition at line 27 of file raytrace.c.

```
00029
00030
         static atm_t atm, atm2;
static ctl_t ctl;
static los_t los;
00031
00032
00034
          static obs_t obs;
00035
00036
         char filename[LEN], losbase[LEN];
00037
00038
         int ig, ip, ir, iw;
00039
00040
         /* Check arguments... */
00041
          if (argc < 4)
00042
            ERRMSG("Give parameters: <ctl> <obs> <atm>");
00043
00044
         /* Read control parameters... */
00045
          read_ctl(argc, argv, &ctl);
00046
          /\star Get basenames... \star/
00047
         scan_ctl(argc, argv, "LOSBASE", -1, "los", losbase);
00048
00049
00050
          /* Read observation geometry... */
00051
         read_obs(NULL, argv[2], &ctl, &obs);
00052
00053
          /★ Read atmospheric data... ★/
00054
          read_atm(NULL, argv[3], &ctl, &atm);
00055
         /* Loop over rays... */
for (ir = 0; ir < obs.nr; ir++) {</pre>
00056
00057
00058
00059
            /* Raytracing... */
00060
            raytrace(&ctl, &atm, &obs, &los, ir);
00061
00062
            /* Copy data... */
00063
            atm2.np = los.np;
            for (ip = 0; ip < los.np; ip++) {
  atm2.time[ip] = obs.time[ir];
  atm2.z[ip] = los.z[ip];</pre>
00064
00065
00066
              atm2.lon[ip] = los.lon[ip];
atm2.lat[ip] = los.lat[ip];
00067
00068
              atm2.p[ip] = los.p[ip];
atm2.t[ip] = los.t[ip];
00069
00071
              for (ig = 0; ig < ctl.ng; ig++)</pre>
              atm2.q[ig][ip] = los.q[ig][ip];
for (iw = 0; iw < ctl.nw; iw++)
atm2.k[iw][ip] = los.k[iw][ip];
00072
00073
00074
00075
            /* Save data... */
00077
            sprintf(filename, "los.%d", ir);
write_atm(NULL, filename, &ctl, &atm2);
00078
00079
08000
00081
00082
         return EXIT_SUCCESS;
00083 }
```

5.26 raytrace.c 251

Here is the call graph for this function:



# 5.26 raytrace.c

```
00001 /*
00002
        This file is part of JURASSIC.
00003
00004
        JURASSIC is free software: you can redistribute it and/or modify
00005
         it under the terms of the GNU General Public License as published by
00006
        the Free Software Foundation, either version 3 of the License, or
00007
         (at your option) any later version.
80000
00009
        JURASSIC is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00010
00011
00012
        GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028
        int argc,
00029
        char *argv[]) {
00030
00031
        static atm_t atm, atm2;
        static ctl_t ctl;
static los_t los;
00032
00033
00034
        static obs_t obs;
00035
        char filename[LEN], losbase[LEN];
00036
00037
00038
        int ig, ip, ir, iw;
00039
00040
         /* Check arguments... */
00041
          ERRMSG("Give parameters: <ctl> <obs> <atm>");
00042
00043
00044
        /* Read control parameters... */
00045
        read_ctl(argc, argv, &ctl);
00046
```

```
/* Get basenames... */
        scan_ctl(argc, argv, "LOSBASE", -1, "los", losbase);
00048
00049
00050
        /\star Read observation geometry... \star/
00051
        read_obs(NULL, argv[2], &ctl, &obs);
00052
00053
        /* Read atmospheric data... */
00054
        read_atm(NULL, argv[3], &ctl, &atm);
00055
00056
        /* Loop over rays... */
00057
        for (ir = 0; ir < obs.nr; ir++) {</pre>
00058
00059
           /* Raytracing... */
00060
           raytrace(&ctl, &atm, &obs, &los, ir);
00061
00062
           /* Copy data... */
00063
           atm2.np = los.np;
          for (ip = 0; ip < los.np; ip++) {
  atm2.time[ip] = obs.time[ir];</pre>
00064
00065
             atm2.z[ip] = los.z[ip];
00066
            atm2.lon[ip] = los.lon[ip];
atm2.lat[ip] = los.lat[ip];
00067
00068
            atm2.p[ip] = los.p[ip];
atm2.t[ip] = los.t[ip];
00069
00070
00071
            for (ig = 0; ig < ctl.ng; ig++)</pre>
00072
               atm2.q[ig][ip] = los.q[ig][ip];
00073
             for (iw = 0; iw < ctl.nw; iw++)
00074
               atm2.k[iw][ip] = los.k[iw][ip];
00075
          }
00076
          /* Save data... */
00077
          sprintf(filename, "los.%d", ir);
00079
           write_atm(NULL, filename, &ctl, &atm2);
08000
00081
        return EXIT_SUCCESS;
00082
00083 }
```

#### 5.27 retrieval.c File Reference

JURASSIC retrieval processor.

#### **Data Structures**

struct ret t

Retrieval control parameters.

## **Functions**

- void analyze\_avk (ret\_t \*ret, ctl\_t \*ctl, atm\_t \*atm, int \*iqa, int \*ipa, gsl\_matrix \*avk)
   Compute information content and resolution.
- void analyze\_avk\_quantity (gsl\_matrix \*avk, int iq, int \*ipa, size\_t \*n0, size\_t \*n1, double \*cont, double \*res)

  Analyze averaging kernels for individual retrieval target.
- double cost\_function (gsl\_vector \*dx, gsl\_vector \*dy, gsl\_matrix \*s\_a\_inv, gsl\_vector \*sig\_eps\_inv)

  Compute cost function.
- void matrix\_invert (gsl\_matrix \*a)

Invert symmetric matrix.

• void matrix\_product (gsl\_matrix \*a, gsl\_vector \*b, int transpose, gsl\_matrix \*c)

Compute matrix product A^TBA or ABA^T for diagonal matrix B.

- $\bullet \ \ void\ optimal\_estimation\ (ret\_t\ *ret,\ ctl\_t\ *ctl,\ obs\_t\ *obs\_meas,\ obs\_t\ *obs\_i,\ atm\_t\ *atm\_apr,\ atm\_t\ *atm\_i)$
- Carry out optimal estimation retrieval.

   void read ret (int argc, char \*argv[], ctl t \*ctl, ret t \*ret)

Read retrieval control parameters.

• void set\_cov\_apr (ret\_t \*ret, ctl\_t \*ctl, atm\_t \*atm, int \*iqa, int \*ipa, gsl\_matrix \*s\_a)

Set a priori covariance.

void set\_cov\_meas (ret\_t \*ret, ctl\_t \*ctl, obs\_t \*obs, gsl\_vector \*sig\_noise, gsl\_vector \*sig\_formod, gsl\_
 vector \*sig\_eps\_inv)

Set measurement errors.

void write\_stddev (const char \*quantity, ret\_t \*ret, ctl\_t \*ctl, atm\_t \*atm, gsl\_matrix \*s)

Write retrieval error to file.

int main (int argc, char \*argv[])

#### 5.27.1 Detailed Description

JURASSIC retrieval processor.

Definition in file retrieval.c.

### 5.27.2 Function Documentation

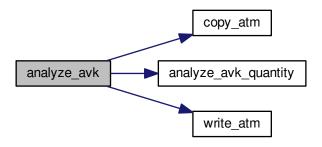
```
5.27.2.1 void analyze_avk ( ret_t * ret, ctl_t * ctl, atm_t * atm, int * iqa, int * ipa, gsl_matrix * avk )
```

Compute information content and resolution.

Definition at line 236 of file retrieval.c.

```
00242
00243
00244
         static atm_t atm_cont, atm_res;
00246
         int ig, iq, iw;
00247
00248
         size_t i, n, n0[NQ], n1[NQ];
00249
00250
         /* Get sizes... */
00251
         n = avk -> size1;
00252
00253
         /\star Find sub-matrices for different quantities... \star/
         for (iq = 0; iq < NQ; iq++) {
    n0[iq] = N;
    for (i = 0; i < n; i++) {
        if (iqa[i] == iq && n0[iq] == N)
00254
00255
00256
00257
00258
                n0[iq] = i;
              if (iqa[i] == iq)
n1[iq] = i - n0[iq] + 1;
00259
00260
00261
            }
00262
00263
00264
         /* Initialize... */
00265
         copy_atm(ctl, &atm_cont, atm, 1);
00266
         copy_atm(ctl, &atm_res, atm, 1);
00267
        /* Analyze quantities... */
analyze_avk_quantity(avk, IDXP, ipa, n0, n1, atm_cont.p, atm_res.
00268
00269
p);
          analyze_avk_quantity(avk, IDXT, ipa, n0, n1, atm_cont.t, atm_res.
00271
         for (ig = 0; ig < ctl->ng; ig++)
00272
           analyze_avk_quantity(avk, IDXQ(ig), ipa, n0, n1,
00273
                                     atm_cont.q[ig], atm_res.q[ig]);
00274
         for (iw = 0; iw < ctl->nw; iw++)
00275
          analyze_avk_quantity(avk, IDXK(iw), ipa, n0, n1,
00276
                                     atm_cont.k[iw], atm_res.k[iw]);
00277
00278
       /* Write results to disk... */
write_atm(ret->dir, "atm_cont.tab", ctl, &atm_cont);
write_atm(ret->dir, "atm_res.tab", ctl, &atm_res);
00279
00280
00281 }
```

Here is the call graph for this function:



5.27.2.2 void analyze\_avk\_quantity ( gsl\_matrix \* avk, int iq, int \* ipa, size\_t \* n0, size\_t \* n1, double \* cont, double \* res )

Analyze averaging kernels for individual retrieval target.

Definition at line 285 of file retrieval.c.

```
00293
00294
           size_t i, j;
00295
           /* Loop over state vector elements... */
if (n0[iq] < N)
    for (i = 0; i < n1[iq]; i++) {</pre>
00296
00297
00299
00300
                  /\star Get area of averagig kernel... \star/
                 for (j = 0; j < n1[iq]; j++)
  cont[ipa[n0[iq] + i]] += gsl_matrix_get(avk, n0[iq] + i, n0[iq] + j);</pre>
00301
00302
00303
                /* Get information density... */    res[ipa[n0[iq] + i]] = 1 / gsl_matrix_get(avk, n0[iq] + i, n0[iq] + i);
00304
00305
00306
00307 }
```

5.27.2.3 double cost\_function (  $gsl_vector * dx$ ,  $gsl_vector * dy$ ,  $gsl_matrix * s_a_inv$ ,  $gsl_vector * sig_eps_inv$  )

Compute cost function.

Definition at line 311 of file retrieval.c.

```
00315
00316
00317
        qsl_vector *x_aux, *y_aux;
00318
00319
        double chisq_a, chisq_m = 0;
00320
00321
        size_t i, m, n;
00322
       /* Get sizes... */
00323
00324
        m = dy->size;
00325
        n = dx -> size;
00326
00327
        /* Allocate... */
00328
        x_aux = gsl_vector_alloc(n);
00329
       y_aux = gsl_vector_alloc(m);
00330
00331
        /\star Determine normalized cost function...
```

```
(chi^2 = 1/m * [dy^T * S_eps^{-1}] * dy + dx^T * S_a^{-1}] * dx]) */
00333
        for (i = 0; i < m; i++)
         chisq_m +=
00334
            gsl_pow_2(gsl_vector_get(dy, i) * gsl_vector_get(sig_eps_inv, i));
00335
        gsl_blas_dgemv(CblasNoTrans, 1.0, s_a_inv, dx, 0.0, x_aux);
gsl_blas_ddot(dx, x_aux, &chisq_a);
00336
00337
00338
00339
00340
        gsl_vector_free(x_aux);
00341
        gsl_vector_free(y_aux);
00342
00343
        /* Return cost function value... */
00344
        return (chisq_m + chisq_a) / (double) m;
00345 }
```

## 5.27.2.4 void matrix\_invert ( gsl\_matrix \* a )

Invert symmetric matrix.

Definition at line 349 of file retrieval.c.

```
00350
                           {
00351
00352
        size_t diag = 1, i, j, n;
00353
00354
        /* Get size... */
00355
        n = a -> size1;
00357
        /* Check if matrix is diagonal... */
        for (i = 0; i < n && diag; i++)
  for (j = i + 1; j < n; j++)
    if (gsl_matrix_get(a, i, j) != 0) {</pre>
00358
00359
00360
00361
              diag = 0;
00362
               break;
00363
00364
00365
        /\star Quick inversion of diagonal matrix... \star/
        if (diag)
for (i = 0; i < n; i++)
00366
00367
             gsl_matrix_set(a, i, i, 1 / gsl_matrix_get(a, i, i));
00368
00370
        /\star Matrix inversion by means of Cholesky decomposition... \star/
        00371
00372
00373
          gsl_linalg_cholesky_invert(a);
00374
00375 }
```

5.27.2.5 void matrix\_product (  $gsl_matrix * a$ ,  $gsl_vector * b$ , int transpose,  $gsl_matrix * c$  )

Compute matrix product A^TBA or ABA^T for diagonal matrix B.

Definition at line 379 of file retrieval.c.

```
00384
00385
        gsl_matrix *aux;
00386
00387
        size_t i, j, m, n;
00388
00389
        /* Set sizes... */
00390
        m = a -> size1;
00391
        n = a -> size2;
00392
00393
        /* Allocate... */
00394
        aux = gsl_matrix_alloc(m, n);
00395
00396
        /* Compute A^T B A... */
00397
        if (transpose == 1) {
00398
          /* Compute B^1/2 A... */
for (i = 0; i < m; i++)
00399
00400
00401
            for (j = 0; j < n; j++)
00402
               gsl_matrix_set(aux, i, j,
```

```
00403
                              gsl_vector_get(b, i) * gsl_matrix_get(a, i, j));
00404
00405
           /* Compute A^T B A = (B^1/2 A)^T (B^1/2 A)...*/
00406
          gsl_blas_dgemm(CblasTrans, CblasNoTrans, 1.0, aux, aux, 0.0, c);
00407
00408
00409
        /* Compute A B A^T... */
00410
        else if (transpose == 2) {
00411
00412
          /* Compute A B^1/2... */
          for (i = 0; i < m; i++)
for (j = 0; j < n; j++)
00413
00414
00415
              gsl_matrix_set(aux, i, j,
00416
                              gsl_matrix_get(a, i, j) * gsl_vector_get(b, j));
00417
          /* Compute A B A^T = (A B^1/2) (A B^1/2)^T... */
00418
00419
          gsl_blas_dgemm(CblasNoTrans, CblasTrans, 1.0, aux, aux, 0.0, c);
00420
00422
        /* Free... */
00423
        gsl_matrix_free(aux);
00424 }
```

5.27.2.6 void optimal\_estimation ( ret\_t \* ret, ctl\_t \* ctl, obs\_t \* obs\_meas, obs\_t \* obs\_i, atm\_t \* atm\_apr, atm\_t \*  $atm_i$ )

Carry out optimal estimation retrieval.

Definition at line 428 of file retrieval.c.

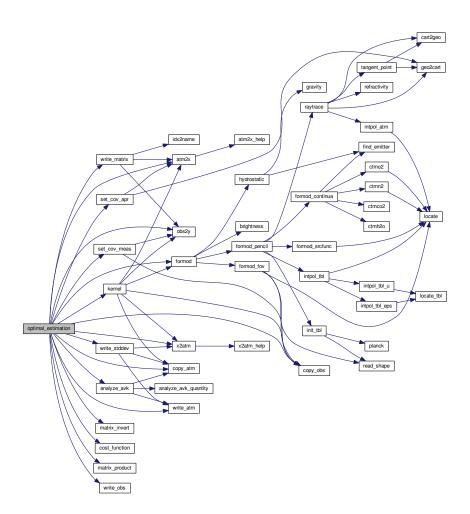
```
00434
00435
00436
       static int ipa[N], iqa[N];
00437
00438
       gsl_matrix *a, *auxnm, *corr, *cov, *gain, *k_i, *s_a_inv;
       gsl_vector *b, *dx, *dy, *sig_eps_inv, *sig_formod, *sig_noise,
00439
00440
          *x_a, *x_i, *x_step, *y_aux, *y_i, *y_m;
00441
00442
       FILE *out:
00443
00444
       char filename[LEN]:
00445
00446
       double chisq, chisq_old, disq = 0, lmpar = 0.001;
00447
00448
       int ig, ip, it = 0, it2, iw;
00449
00450
       size t i, j, m, n;
00451
00452
00453
           Initialize...
00454
00455
00456
       /* Get sizes... */
00457
        m = obs2y(ctl, obs_meas, NULL, NULL, NULL);
00458
        n = atm2x(ctl, atm_apr, NULL, iqa, ipa);
00459
        if (m <= 0 || n <= 0)
00460
          ERRMSG("Check problem definition!");
00461
00462
        /* Write info... */
00463
       printf("Problem size: m= %d / n= %d "
                "(alloc= %.4g MB / stat= %.4g MB)n",
00464
00465
                (int) m, (int) n,
                (double) (3 \star m \star n + 4 \star n \star n + 8 \star m +
00466
               8 * n) * sizeof(double) / 1024. / 1024.,
(double) (5 * sizeof(atm_t) + 3 * sizeof(obs_t)
00467
00468
                          + 2 * N * sizeof(int)) / 1024. / 1024.);
00469
00470
00471
        /* Allocate... */
00472
        a = gsl_matrix_alloc(n, n);
00473
        cov = gsl_matrix_alloc(n, n);
        k_i = gsl_matrix_alloc(m, n);
00474
00475
        s_a_inv = gsl_matrix_alloc(n, n);
00476
00477
        b = gsl_vector_alloc(n);
00478
        dx = gsl_vector_alloc(n);
        dy = gsl_vector_alloc(m);
00479
00480
        sig_eps_inv = gsl_vector_alloc(m);
        sig_formod = gsl_vector_alloc(m);
00481
00482
       sig_noise = gsl_vector_alloc(m);
00483
       x_a = gsl_vector_alloc(n);
```

```
00484
        x_i = gsl_vector_alloc(n);
        x_step = gsl_vector_alloc(n);
y_aux = gsl_vector_alloc(m);
00485
00486
00487
        y_i = gsl_vector_alloc(m);
00488
        y_m = gsl_vector_alloc(m);
00489
00490
        /* Set initial state... */
00491
        copy_atm(ctl, atm_i, atm_apr, 0);
00492
        copy_obs(ctl, obs_i, obs_meas, 0);
00493
        formod(ctl, atm_i, obs_i);
00494
00495
        /* Set state vectors and observation vectors... */
00496
        atm2x(ctl, atm_apr, x_a, NULL, NULL);
00497
        atm2x(ctl, atm_i, x_i, NULL, NULL);
00498
        obs2y(ctl, obs_meas, y_m, NULL, NULL);
00499
        obs2y(ctl, obs_i, y_i, NULL, NULL);
00500
00501
        /* Set inverse a priori covariance S_a^-1... */
        set_cov_apr(ret, ctl, atm_apr, iqa, ipa, s_a_inv);
write_matrix(ret->dir, "matrix_cov_apr.tab", ctl, s_a_inv,
00502
00503
00504
                      atm_i, obs_i, "x", "x", "r");
00505
        matrix_invert(s_a_inv);
00506
00507
        /* Get measurement errors... */
00508
        set_cov_meas(ret, ctl, obs_meas, siq_noise, siq_formod, siq_eps_inv);
00509
        /* Create cost function file... */
00510
00511
        sprintf(filename, "%s/costs.tab", ret->dir);
00512
        if (!(out = fopen(filename, "w")))
          ERRMSG("Cannot create cost function file!");
00513
00514
00515
         /* Write header... */
00516
        fprintf(out,
00517
                 "# $1 = iteration number n"
                 "# $2 = normalized cost function\n"
00518
                 "# $3 = number of measurements\n'
00519
                 "# $4 = number of state vector elements\n\n");
00520
00522
        /* Determine dx = x_i - x_a and dy = y - F(x_i) ... */
00523
        gsl_vector_memcpy(dx, x_i);
00524
        gsl_vector_sub(dx, x_a);
00525
        gsl_vector_memcpy(dy, y_m);
00526
        gsl_vector_sub(dy, y_i);
00527
00528
         /* Compute cost function... */
00529
        chisq = cost_function(dx, dy, s_a_inv, sig_eps_inv);
00530
        /* Write info... */
printf("it= %d / chi^2/m= %g\n", it, chisq);
00531
00532
00533
00534
        /* Write to cost function file... */
00535
        fprintf(out, "%d %g %d %d\n", it, chisq, (int) m, (int) n);
00536
00537
        /* Compute initial kernel... */
00538
        kernel(ctl, atm_i, obs_i, k_i);
00539
00540
00541
           Levenberg-Marquardt minimization...
00542
00543
00544
        /* Outer loop... */
        for (it = 1; it <= ret->conv_itmax; it++) {
00545
00546
00547
           /* Store current cost function value... */
00548
          chisq_old = chisq;
00549
00550
          /* Compute kernel matrix K_{i}... */
00551
          if (it > 1 && it % ret->kernel recomp == 0)
00552
            kernel(ctl, atm_i, obs_i, k_i);
00554
          /* Compute K_i^T * S_eps^{-1} * K_i ... */
if (it == 1 || it % ret->kernel_recomp == 0)
00555
00556
            matrix_product(k_i, sig_eps_inv, 1, cov);
00557
00558
           /* Determine b = K_i^T * S_eps^{-1} * dy - S_a^{-1} * dx ... */
00559
          for (i = 0; i < m; i++)
00560
            gsl_vector_set(y_aux, i, gsl_vector_get(dy, i)
00561
                             * gsl_pow_2(gsl_vector_get(sig_eps_inv, i)));
          {\tt gsl\_blas\_dgemv(CblasTrans, 1.0, k\_i, y\_aux, 0.0, b);}
00562
00563
          gsl_blas_dgemv(CblasNoTrans, -1.0, s_a_inv, dx, 1.0, b);
00564
00565
           /* Inner loop... */
00566
          for (it2 = 0; it2 < 20; it2++) {
00567
00568
             /* Compute A = (1 + lmpar) * S_a^{-1} + K_i^T * S_eps^{-1} * K_i ... */
            gsl_matrix_memcpy(a, s_a_inv);
gsl_matrix_scale(a, 1 + lmpar);
00569
00570
```

```
gsl_matrix_add(a, cov);
00572
00573
              /* Solve A * x_step = b by means of Cholesky decomposition... */
00574
              gsl_linalg_cholesky_decomp(a);
00575
             gsl_linalg_cholesky_solve(a, b, x_step);
00576
00577
              /* Update atmospheric state... */
00578
             gsl_vector_add(x_i, x_step);
00579
              copy_atm(ctl, atm_i, atm_apr, 0);
00580
              copy_obs(ctl, obs_i, obs_meas, 0);
00581
             x2atm(ctl, x_i, atm_i);
00582
00583
              /* Check atmospheric state... */
00584
              for (ip = 0; ip < atm_i->np; ip++) {
00585
                atm_i \rightarrow p[ip] = GSL_MIN(GSL_MAX(atm_i \rightarrow p[ip], 5e-7), 5e4);
                atm_i \rightarrow t[ip] = GSL_MIN(GSL_MAX(atm_i \rightarrow t[ip], 100), 400);
00586
                for (ig = 0; ig < ctl->ng; ig++)
  atm_i->q[ig][ip] = GSL_MIN(GSL_MAX(atm_i->q[ig][ip], 0), 1);
for (iw = 0; iw < ctl->nw; iw++)
00587
00588
00589
00590
                 atm_i \rightarrow k[iw][ip] = GSL_MAX(atm_i \rightarrow k[iw][ip], 0);
00591
00592
00593
              /* Forward calculation... */
00594
             formod(ctl, atm_i, obs_i);
obs2y(ctl, obs_i, y_i, NULL, NULL);
00595
00596
              /* Determine dx = x_i - x_a and dy = y - F(x_i) ... \star/
00597
00598
              gsl_vector_memcpy(dx, x_i);
00599
              gsl_vector_sub(dx, x_a);
00600
              gsl_vector_memcpy(dy, y_m);
00601
             gsl_vector_sub(dy, y_i);
00602
00603
              /* Compute cost function... */
00604
             chisq = cost_function(dx, dy, s_a_inv, sig_eps_inv);
00605
              /* Modify Levenberg-Marquardt parameter... */
00606
             if (chisq > chisq_old) {
  lmpar *= 10;
00607
00608
00609
                gsl_vector_sub(x_i, x_step);
00610
               lmpar /= 10;
00611
00612
               break;
00613
00614
           }
00615
00616
           /* Write info... */
00617
           printf("it= %d / chi^2/m= %g\n", it, chisq);
00618
           /* Write to cost function file... */
00619
           fprintf(out, "%d %g %d %d\n", it, chisq, (int) m, (int) n);
00620
00621
00622
           /* Get normalized step size in state space... */
00623
           gsl_blas_ddot(x_step, b, \&disq);
00624
           disq /= (double) n;
00625
00626
           /* Convergence test... */
           if ((it == 1 || it % ret->kernel_recomp == 0) && disq < ret->
      conv_dmin)
00628
00629
00630
00631
         /* Close cost function file... */
00632
         fclose(out);
00633
         /\star Store results... \star/
00634
        write_atm(ret->dir, "atm_final.tab", ctl, atm_i);
write_obs(ret->dir, "obs_final.tab", ctl, obs_i);
write_matrix(ret->dir, "matrix_kernel.tab", ctl, k_i,
00635
00636
00637
                       atm_i, obs_i, "y", "x", "r");
00638
00639
00640
00641
           Analysis of retrieval results...
00642
00643
00644
         /* Check if error analysis is requested... */
         if (ret->err_ana) {
00645
00646
00647
           /* Allocate... */
00648
           auxnm = gsl_matrix_alloc(n, m);
00649
           corr = gsl_matrix_alloc(n, n);
           gain = gsl_matrix_alloc(n, m);
00650
00651
00652
           /\star Compute inverse retrieval covariance...
00653
              cov^{-1} = S_a^{-1} + K_i^T * S_eps^{-1} * K_i */
           matrix_product(k_i, sig_eps_inv, 1, cov);
00654
00655
           gsl_matrix_add(cov, s_a_inv);
00656
```

```
00657
          /* Compute retrieval covariance... */
00658
          matrix_invert(cov);
          write_matrix(ret->dir, "matrix_cov_ret.tab", ctl, cov,
00659
          atm_i, obs_i, "x", "x", "r");
write_stddev("total", ret, ctl, atm_i, cov);
00660
00661
00662
00663
           /* Compute correlation matrix... */
00664
          for (i = 0; i < n; i++)
00665
           for (j = 0; j < n; j++)
00666
              gsl_matrix_set(corr, i, j, gsl_matrix_get(cov, i, j)
00667
                               / sqrt(gsl_matrix_get(cov, i, i))
          / sqrt(gsl_matrix_get(cov, j, j)));
write_matrix(ret->dir, "matrix_corr.tab", ctl, corr,
00668
00669
00670
                        atm_i, obs_i, "x", "x", "r");
00671
00672
          /* Compute gain matrix..
          G = cov * K^T * S_eps^{-1} */
for (i = 0; i < n; i++)
00673
00674
            for (j = 0; j < m; j++)
00675
              gsl_matrix_set(auxnm, i, j, gsl_matrix_get(k_i, j, i)
    * gsl_pow_2(gsl_vector_get(sig_eps_inv, j)));
00676
00677
          00678
00679
00680
00681
00682
          /\star Compute retrieval error due to noise... \star/
          matrix_product(gain, sig_noise, 2, a);
write_stddev("noise", ret, ctl, atm_i, a);
00683
00684
00685
00686
          /* Compute retrieval error due to forward model errors... */
00687
          matrix_product(gain, sig_formod, 2, a);
00688
          write_stddev("formod", ret, ctl, atm_i, a);
00689
00690
          /\star Compute averaging kernel matrix
             A = G * K ...
00691
          gsl_blas_dgemm(CblasNoTrans, CblasNoTrans, 1.0, gain, k_i, 0.0, a);
00692
          write_matrix(ret->dir, "matrix_avk.tab", ctl, a, atm_i, obs_i, "x", "x", "r");
00693
00694
00695
00696
          /* Analyze averaging kernel matrix... */
00697
          analyze_avk(ret, ctl, atm_i, iqa, ipa, a);
00698
00699
          /* Free... */
00700
          gsl_matrix_free(auxnm);
00701
          gsl_matrix_free(corr);
00702
          gsl_matrix_free(gain);
00703
00704
00705
00706
           Finalize...
00707
00708
00709
        gsl_matrix_free(a);
00710
        gsl_matrix_free(cov);
00711
        gsl_matrix_free(k_i);
00712
        gsl_matrix_free(s_a_inv);
00713
00714
        gsl_vector_free(b);
00715
        gsl_vector_free(dx);
00716
        gsl_vector_free(dy);
00717
        gsl_vector_free(sig_eps_inv);
00718
        qsl_vector_free(sig_formod);
00719
        gsl_vector_free(sig_noise);
00720
        gsl_vector_free(x_a);
00721
        gsl_vector_free(x_i);
00722
        gsl_vector_free(x_step);
00723
        gsl_vector_free(y_aux);
00724
        gsl_vector_free(y_i);
00725
        gsl_vector_free(y_m);
00726 }
```

Here is the call graph for this function:



5.27.2.7 void read\_ret ( int argc, char \* argv[], ctl\_t \* ctl, ret\_t \* ret )

Read retrieval control parameters.

Definition at line 730 of file retrieval.c.

```
00734
00735
00736
          int id, ig, iw;
00737
00738
          /* Iteration control... */
00739
          ret->kernel_recomp =
          (int) scan_ctl(argc, argv, "KERNEL_RECOMP", -1, "3", NULL);
ret->conv_itmax = (int) scan_ctl(argc, argv, "CONV_ITMAX", -1, "30", NULL);
ret->conv_dmin = scan_ctl(argc, argv, "CONV_DMIN", -1, "0.1", NULL);
00740
00741
00742
00743
00744
           /* Error analysis... */
          ret->err_ana = (int) scan_ctl(argc, argv, "ERR_ANA", -1, "1", NULL);
00745
00746
00747
           for (id = 0; id < ctl->nd; id++)
00748
             ret->err_formod[id] = scan_ctl(argc, argv, "ERR_FORMOD", id, "0", NULL);
00749
00750
          for (id = 0; id < ctl->nd; id++)
             ret->err_noise[id] = scan_ctl(argc, argv, "ERR_NOISE", id, "0", NULL);
00751
00752
          ret->err_press = scan_ctl(argc, argv, "ERR_PRESS", -1, "0", NULL);
ret->err_press_cz = scan_ctl(argc, argv, "ERR_PRESS_CZ", -1, "-999", NULL);
ret->err_press_ch = scan_ctl(argc, argv, "ERR_PRESS_CH", -1, "-999", NULL);
00753
00754
00755
```

```
00756
                 ret->err_temp = scan_ctl(argc, argv, "ERR_TEMP", -1, "0", NULL);
ret->err_temp_cz = scan_ctl(argc, argv, "ERR_TEMP_CZ", -1, "-999", NULL);
ret->err_temp_ch = scan_ctl(argc, argv, "ERR_TEMP_CH", -1, "-999", NULL);
00757
00758
00759
00760
                 for (ig = 0; ig < ctl->ng; ig++) {
  ret->err_q[ig] = scan_ctl(argc, argv, "ERR_Q", ig, "0", NULL);
  ret->err_q_cz[ig] = scan_ctl(argc, argv, "ERR_Q_CZ", ig, "-999", NULL);
  ret->err_q_ch[ig] = scan_ctl(argc, argv, "ERR_Q_CH", ig, "-999", NULL);
00761
00762
00763
00764
00765
00766
                 for (iw = 0; iw < ctl->nw; iw++) {
00767
                  ret->err_k[iw] = scan_ctl(argc, argv, "ERR_K", iw, "0", NULL);
ret->err_k_cz[iw] = scan_ctl(argc, argv, "ERR_K_CZ", iw, "-999", NULL);
ret->err_k_ch[iw] = scan_ctl(argc, argv, "ERR_K_CH", iw, "-999", NULL);
00768
00769
00770
00771
00772 }
```

Here is the call graph for this function:



```
5.27.2.8 void set_cov_apr ( ret_t * ret, ctl_t * ctl, atm_t * atm, int * iqa, int * ipa, gsl_matrix * s_a)
```

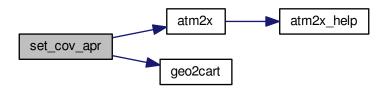
Set a priori covariance.

Definition at line 776 of file retrieval.c.

```
00782
00783
00784
        qsl vector *x a;
00785
00786
        double ch, cz, rho, x0[3], x1[3];
00787
00788
        int ig, iw;
00789
00790
        size_t i, j, n;
00791
00792
        /* Get sizes... */
00793
        n = s_a->size1;
00794
00795
        /* Allocate... */
00796
        x_a = gsl_vector_alloc(n);
00797
00798
        /* Get sigma vector...
00799
        atm2x(ct1, atm, x_a, NULL, NULL);
00800
        for (i = 0; i < n; i++) {</pre>
         if (iqa[i] == IDXP)
00801
          gsl_vector_set(x_a, i, ret->err_press / 100 * gsl_vector_get(x_a, i));
if (iqa[i] == IDXT)
00802
00803
00804
             gsl_vector_set(x_a, i, ret->err_temp);
          for (ig = 0; ig < ctl->ng; ig++)
  if (iqa[i] == IDXQ(ig))
00805
00806
00807
               gsl_vector_set(x_a, i, ret->err_q[ig] / 100 * gsl_vector_get(x_a, i));
          for (iw = 0; iw < ctl->nw; iw++)
  if (iqa[i] == IDXK(iw))
00808
00809
00810
               gsl_vector_set(x_a, i, ret->err_k[iw]);
00811
00812
00813
        /\star Check standard deviations... \star/
        for (i = 0; i < n; i++)
  if (gsl_pow_2(gsl_vector_get(x_a, i)) <= 0)</pre>
00814
00815
00816
             ERRMSG("Check a priori data (zero standard deviation)!");
00817
```

```
/* Initialize diagonal covariance... */
00819
         gsl_matrix_set_zero(s_a);
00820
         for (i = 0; i < n; i++)
00821
           gsl_matrix_set(s_a, i, i, gsl_pow_2(gsl_vector_get(x_a, i)));
00822
00823
         /* Loop over matrix elements... */
         for (i = 0; i < n; i++)
for (j = 0; j < n; j++)
00824
00825
00826
             if (i != j && iqa[i] == iqa[j]) {
00827
00828
                /* Initialize... */
00829
                cz = ch = 0;
00830
00831
                 /* Set correlation lengths for pressure... */
00832
                if (iqa[i] == IDXP) {
                  cz = ret->err_press_cz;
ch = ret->err_press_ch;
00833
00834
                }
00835
00836
00837
                /* Set correlation lengths for temperature... */
00838
                if (iqa[i] == IDXT) {
00839
                  cz = ret->err_temp_cz;
                  ch = ret->err_temp_ch;
00840
00841
00842
00843
                /\star Set correlation lengths for volume mixing ratios... \star/
00844
                for (ig = 0; ig < ctl->ng; ig++)
00845
                 if (iqa[i] == IDXQ(ig)) {
00846
                    cz = ret->err_q_cz[ig];
                    ch = ret->err_q_ch[ig];
00847
00848
00849
00850
                /* Set correlation lengths for extinction... */
                for (iw = 0; iw < ctl->nw; iw++)
  if (iqa[i] == IDXK(iw)) {
    cz = ret->err_k_cz[iw];
00851
00852
00853
00854
                     ch = ret->err_k_ch[iw];
00855
00856
00857
                /* Compute correlations... */
00858
                if (cz > 0 && ch > 0) {
00859
                  /* Get Cartesian coordinates... */
geo2cart(0, atm->lon[ipa[i]], atm->lat[ipa[i]], x0);
geo2cart(0, atm->lon[ipa[j]], atm->lat[ipa[j]], x1);
00860
00861
00862
00863
00864
                  /* Compute correlations... */
00865
                  rho =
00866
                     exp(-DIST(x0, x1) / ch -
                          fabs(atm->z[ipa[i]] - atm->z[ipa[j]]) / cz);
00867
00868
                  /* Set covariance... */
gsl_matrix_set(s_a, i, j, gsl_vector_get(x_a, i)
00869
00870
00871
                                    * gsl_vector_get(x_a, j) * rho);
00872
00873
              }
00875
         /* Free... */
00876
         gsl_vector_free(x_a);
00877 }
```

Here is the call graph for this function:



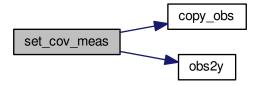
5.27.2.9 void set\_cov\_meas ( ret\_t \* ret, ctl\_t \* ctl, obs\_t \* obs, gsl\_vector \* sig\_noise, gsl\_vector \* sig\_eps\_inv )

Set measurement errors.

Definition at line 881 of file retrieval.c.

```
00887
00888
00889
        static obs_t obs_err;
00890
00891
        int id, ir;
00892
00893
        size_t i, m;
00894
00895
        /* Get size... */
00896
        m = sig_eps_inv->size;
00897
00898
        /\star Noise error (always considered in retrieval fit)... \star/
00899
        copy_obs(ctl, &obs_err, obs, 1);
00900
        for (ir = 0; ir < obs_err.nr; ir++)
  for (id = 0; id < ctl->nd; id++)
00901
00902
             obs_err.rad[id][ir]
00903
               = (gsl_finite(obs->rad[id][ir]) ? ret->err_noise[id] : GSL_NAN);
00904
        obs2y(ctl, &obs_err, sig_noise, NULL, NULL);
00905
00906
        /\star Forward model error (always considered in retrieval fit)... \star/
00907
        copy_obs(ctl, &obs_err, obs, 1);
for (ir = 0; ir < obs_err.nr; ir++)</pre>
00908
00909
          for (id = 0; id < ctl->nd; id++)
00910
             obs_err.rad[id][ir]
00911
               = fabs(ret->err_formod[id] / 100 * obs->rad[id][ir]);
        obs2y(ctl, &obs_err, sig_formod, NULL, NULL);
00912
00913
        /* Total error... */
00915
        for (i = 0; i < m; i++)</pre>
00916
          gsl_vector_set(sig_eps_inv, i,
                           1 / sqrt(gsl_pow_2(gsl_vector_get(sig_noise, i))
00917
00918
                                     + gsl_pow_2(gsl_vector_get(sig_formod, i))));
00919
00920
        /* Check standard deviations... */
        for (i = 0; i < m; i++)</pre>
00922
          if (gsl_vector_get(sig_eps_inv, i) <= 0)</pre>
00923
             ERRMSG("Check measurement errors (zero standard deviation)!");
00924 }
```

Here is the call graph for this function:



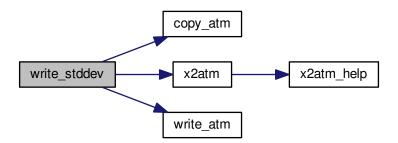
5.27.2.10 void write\_stddev ( const char \* quantity, ret\_t \* ret, ctl\_t \* ctl, atm\_t \* atm, gsl\_matrix \* s )

Write retrieval error to file.

Definition at line 928 of file retrieval.c.

```
00933
00934
00935
         static atm_t atm_aux;
00936
00937
         gsl_vector *x_aux;
00938
00939
        char filename[LEN];
00940
00941
         size_t i, n;
00942
         /* Get sizes... */
00943
00944
         n = s \rightarrow size1:
00945
00946
         /* Allocate... */
00947
         x_aux = gsl_vector_alloc(n);
00948
00949
         /\star Compute standard deviation... \star/
00950
         for (i = 0; i < n; i++)
00951
           gsl_vector_set(x_aux, i, sqrt(gsl_matrix_get(s, i, i)));
00952
00953
         /* Write to disk... */
00954
         copy_atm(ctl, &atm_aux, atm, 1);
        x2atm(ctl, x_aux, &atm_aux);
sprintf(filename, "atm_err_%s.tab", quantity);
write_atm(ret->dir, filename, ctl, &atm_aux);
00955
00956
00957
00958
00959
00960
        gsl_vector_free(x_aux);
00961 }
```

Here is the call graph for this function:



## 5.27.2.11 int main ( int argc, char \* argv[] )

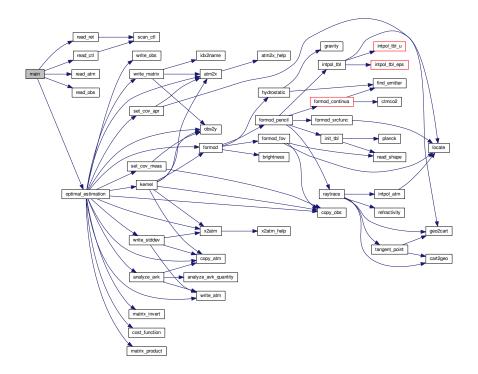
Definition at line 180 of file retrieval.c.

```
00182
                          {
00183
        static atm_t atm_i, atm_apr;
static ctl_t ctl;
static obs_t obs_i, obs_meas;
00184
00185
00186
00187
         static ret_t ret;
00188
         FILE *dirlist;
00189
00190
00191
         /\!\star Check arguments... \star/
00192
         if (argc < 3)
00193
           ERRMSG("Give parameters: <ctl> <dirlist>");
00194
00195
         /* Measure CPU-time... */
00196
        TIMER("total", 1);
00197
00198
        /* Read control parameters... */
00199
        read_ctl(argc, argv, &ctl);
```

5.28 retrieval.c 265

```
00200
         read_ret(argc, argv, &ctl, &ret);
00201
         /* Open directory list... */
if (!(dirlist = fopen(argv[2], "r")))
00202
00203
00204
           ERRMSG("Cannot open directory list!");
00205
         /* Loop over directories... */
while (fscanf(dirlist, "%s", ret.dir) != EOF) {
00206
00207
00208
            /* Write info... */ printf("\nRetrieve in directory s...\n\n", ret.dir);
00209
00210
00211
            /* Read atmospheric data... */
read_atm(ret.dir, "atm_apr.tab", &ctl, &atm_apr);
00212
00213
00214
            /* Read observation data... */
read_obs(ret.dir, "obs_meas.tab", &ctl, &obs_meas);
00215
00216
00217
00218
            /* Run retrieval... */
00219
            optimal_estimation(&ret, &ctl, &obs_meas, &obs_i, &atm_apr, &atm_i);
00220
00221
            /* Measure CPU-time... */
            TIMER("total", 2);
00222
00223
00224
00225
          /* Write info... */
00226
         printf("\nRetrieval done...\n");
00227
         /* Measure CPU-time... */
TIMER("total", 3);
00228
00229
00230
00231
         return EXIT_SUCCESS;
00232 }
```

Here is the call graph for this function:



# 5.28 retrieval.c

```
00001 /* 00002 \qquad \text{This file is part of JURASSIC.} \\ 00003 \\ 00004 \qquad \text{JURASSIC is free software: you can redistribute it and/or modify}
```

```
it under the terms of the GNU General Public License as published by
00006
        the Free Software Foundation, either version 3 of the License, or
00007
         (at your option) any later version.
80000
        JURASSIC is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00009
00010
00011
00012
        GNU General Public License for more details.
00013
00014
        You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
        Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 /* -
00028
         Structs...
00029
00030
00032 typedef struct {
00033
00035
        char dir[LEN];
00036
00038
        int kernel_recomp;
00039
00041
        int conv_itmax;
00042
00044
        double conv_dmin;
00045
00047
        int err ana;
00048
00050
        double err_formod[ND];
00051
00053
        double err_noise[ND];
00054
        double err_press;
00057
00059
        double err_press_cz;
00060
00062
        double err_press_ch;
00063
00065
        double err_temp;
00066
00068
        double err_temp_cz;
00069
00071
        double err_temp_ch;
00072
00074
        double err g[NG]:
00075
00077
        double err_q_cz[NG];
00078
00080
        double err_q_ch[NG];
00081
00083
        double err k[NW];
00084
00086
        double err_k_cz[NW];
00087
00089
        double err_k_ch[NW];
00090
00091 } ret_t;
00092
00093 /*
00094
         Functions...
00095
00096
00098 void analyze_avk(
00099
       ret t * ret.
        ctl_t * ctl,
00100
00101
        atm_t * atm,
00102
        int *iqa,
00103
        int *ipa,
00104
        gsl_matrix * avk);
00105
00107 void analyze_avk_quantity(
00108
        gsl_matrix * avk,
00109
        int iq,
00110
        int *ipa,
        size_t * n0,
size_t * n1,
00111
00112
00113
        double *cont,
00114
        double *res);
00115
00117 double cost_function(
00118 gsl_vector * dx,
00119
       gsl_vector * dy,
```

5.28 retrieval.c 267

```
gsl_matrix * s_a_inv,
gsl_vector * sig_eps_inv);
00121
00122
00124 void matrix_invert(
00125
        gsl_matrix * a);
00126
00128 void matrix_product(
00129
       gsl_matrix * a,
        gsl_vector * b,
00130
        int transpose,
00131
00132
       gsl_matrix * c);
00133
00135 void optimal_estimation(
00136 ret_t * ret,
00137
        ctl_t * ctl,
00138
        obs_t * obs_meas,
        obs_t * obs_i,
00139
        atm_t * atm_apr,
00140
00141
        atm_t * atm_i);
00142
00144 void read_ret(
00145
        int argc,
        char *argv[],
ctl_t * ctl,
ret_t * ret);
00146
00147
00148
00149
00151 void set_cov_apr(
00152 ret_t * ret,
        ctl_t * ctl,
00153
        atm_t * atm,
00154
00155
        int *iqa,
00156
        int *ipa,
00157
        gsl_matrix * s_a);
00158
00160 void set_cov_meas(
        ret_t * ret,
ctl_t * ctl,
00161
00162
        obs_t * obs,
00163
00164
        gsl_vector * sig_noise,
gsl_vector * sig_formod,
gsl_vector * sig_eps_inv);
00165
00166
00167
00169 void write stddev(
00170
        const char *quantity,
00171
        ret_t * ret,
00172
        ctl_t * ctl,
        atm_t * atm,
00173
00174
        gsl_matrix * s);
00175
00176 /* --
00177
        Main...
00178
00179
00180 int main(
00181
       int argc,
00182
        char *argv[]) {
00184
        static atm_t atm_i, atm_apr;
00185
        static ctl_t ctl;
00186
        static obs_t obs_i, obs_meas;
        static ret_t ret;
00187
00188
00189
        FILE *dirlist;
00190
00191
        /* Check arguments... */
00192
        if (argc < 3)</pre>
00193
          ERRMSG("Give parameters: <ctl> <dirlist>");
00194
00195
        /* Measure CPU-time... */
00196
        TIMER("total", 1);
00197
00198
        /* Read control parameters... ∗/
00199
        read_ctl(argc, argv, &ctl);
00200
        read_ret(argc, argv, &ctl, &ret);
00201
00202
        /* Open directory list... */
00203
        if (!(dirlist = fopen(argv[2], "r")))
00204
          ERRMSG("Cannot open directory list!");
00205
        /* Loop over directories... */
while (fscanf(dirlist, "%s", ret.dir) != EOF) {
00206
00207
00208
          /* Write info... */
printf("\nRetrieve in directory %s...\n\n", ret.dir);
00209
00210
00211
          /* Read atmospheric data... */
read_atm(ret.dir, "atm_apr.tab", &ctl, &atm_apr);
00212
00213
```

```
00214
00215
          /* Read observation data... */
          read_obs(ret.dir, "obs_meas.tab", &ctl, &obs_meas);
00216
00217
00218
          /* Run retrieval... */
00219
          optimal_estimation(&ret, &ctl, &obs_meas, &obs_i, &atm_apr, &atm_i);
00220
00221
          /* Measure CPU-time... */
00222
         TIMER("total", 2);
00223
00224
00225
       /* Write info... */
       printf("\nRetrieval done...\n");
00226
00227
00228
        /* Measure CPU-time... */
00229
       TIMER("total", 3);
00230
00231
       return EXIT SUCCESS;
00232 }
00233
00235
00236 void analyze_avk(
00237
       ret_t * ret,
ctl_t * ctl,
00238
00239
        atm_t * atm,
00240
        int *iqa,
00241
       int *ipa,
00242
       gsl_matrix * avk) {
00243
00244
       static atm_t atm_cont, atm_res;
00245
00246
       int ig, iq, iw;
00247
00248
       size_t i, n, n0[NQ], n1[NQ];
00249
00250
       /* Get sizes... */
       n = avk->size1;
00252
00253
        /* Find sub-matrices for different quantities... */
       for (iq = 0; iq < NQ; iq++) {
  n0[iq] = N;
  for (i = 0; i < n; i++) {</pre>
00254
00255
00256
           if (iqa[i] == iq && n0[iq] == N)
00257
00258
             n0[iq] = i;
00259
            if (iqa[i] == iq)
00260
             n1[iq] = i - n0[iq] + 1;
00261
          }
00262
       }
00263
00264
        /* Initialize... */
00265
        copy_atm(ctl, &atm_cont, atm, 1);
00266
       copy_atm(ctl, &atm_res, atm, 1);
00267
       /* Analyze quantities... */
00268
        analyze_avk_quantity(avk, IDXP, ipa, n0, n1, atm_cont.p, atm_res.
00269
p);
        analyze_avk_quantity(avk, IDXT, ipa, n0, n1, atm_cont.t, atm_res.
     t);
00271
        for (ig = 0; ig < ctl->ng; ig++)
         analyze_avk_quantity(avk, IDXQ(ig), ipa, n0, n1,
00272
00273
                               atm_cont.q[ig], atm_res.q[ig]);
00274
        for (iw = 0; iw < ctl->nw; iw++)
00275
         analyze_avk_quantity(avk, IDXK(iw), ipa, n0, n1,
00276
                               atm_cont.k[iw], atm_res.k[iw]);
00277
       /* Write results to disk... */
write_atm(ret->dir, "atm_cont.tab", ctl, &atm_cont);
write_atm(ret->dir, "atm_res.tab", ctl, &atm_res);
00278
00279
00280
00282
00284
00285 void analyze_avk_quantity(
00286
       gsl matrix * avk,
00287
        int iq,
00288
        int *ipa,
       size_t * n0,
size_t * n1,
00289
00290
00291
       double *cont,
       double *res) {
00292
00293
00294
        size_t i, j;
00295
00296
        /* Loop over state vector elements... */
       if (n0[iq] < N)
  for (i = 0; i < n1[iq]; i++) {</pre>
00297
00298
```

5.28 retrieval.c 269

```
00300
           /* Get area of averagig kernel... */
           for (j = 0; j < n1[iq]; j++)
  cont[ipa[n0[iq] + i]] += gsl_matrix_get(avk, n0[iq] + i, n0[iq] + j);</pre>
00301
00302
00303
          00304
00306
00307 }
00308
00310
00311 double cost_function(
00312
       gsl_vector * dx,
00313
       gsl_vector * dy,
00314
       gsl_matrix * s_a_inv,
       gsl_vector * sig_eps_inv) {
00315
00316
00317
       gsl_vector *x_aux, *y_aux;
00318
00319
       double chisq_a, chisq_m = 0;
00320
00321
       size_t i, m, n;
00322
00323
       /* Get sizes... */
       m = dy->size;
00324
00325
       n = dx -> size;
00326
00327
       /* Allocate... */
00328
       x_aux = gsl_vector_alloc(n);
00329
       y_aux = gsl_vector_alloc(m);
00330
00331
       /* Determine normalized cost function...
00332
         (chi^2 = 1/m * [dy^T * S_eps^{-1}] * dy + dx^T * S_a^{-1} * dx]) */
00333
       for (i = 0; i < m; i++)</pre>
00334
        chisq_m +=
       gsl_pow_2(gsl_vector_get(dy, i) * gsl_vector_get(sig_eps_inv, i));
gsl_blas_dgemv(CblasNoTrans, 1.0, s_a_inv, dx, 0.0, x_aux);
00335
00336
00337
       gsl_blas_ddot(dx, x_aux, &chisq_a);
00338
00339
       /* Free... */
00340
       gsl_vector_free(x_aux);
00341
       gsl_vector_free(y_aux);
00342
00343
       /* Return cost function value... */
00344
       return (chisq_m + chisq_a) / (double) m;
00345 }
00346
00348
00349 void matrix_invert(
00350 gsl_matrix * a) {
00351
00352
       size_t diag = 1, i, j, n;
00353
00354
       /* Get size... */
00355
       n = a -> size1;
00356
00357
       /* Check if matrix is diagonal... */
       for (i = 0; i < n && diag; i++)
for (j = i + 1; j < n; j++)</pre>
00358
00359
          if (gsl_matrix_get(a, i, j) != 0) {
00360
00361
            diag = 0;
00362
            break;
00363
00364
00365
       /* Quick inversion of diagonal matrix... */
       if (diag)
00366
        for (i = 0; i < n; i++)
00367
00368
           gsl_matrix_set(a, i, i, 1 / gsl_matrix_get(a, i, i));
00369
00370
       /\star Matrix inversion by means of Cholesky decomposition... \star/
       else {
00371
        gsl_linalg_cholesky_decomp(a);
00372
00373
         gsl_linalg_cholesky_invert(a);
00374
00375 }
00376
00378
00379 void matrix_product(
00380
       gsl_matrix * a,
       gsl_vector * b,
00381
00382
       int transpose,
00383
       gsl_matrix * c) {
00384
00385
       gsl matrix *aux;
```

```
00386
00387
       size_t i, j, m, n;
00388
       /* Set sizes... */
00389
00390
       m = a -> size1;
00391
       n = a \rightarrow size2;
00392
00393
00394
       aux = gsl_matrix_alloc(m, n);
00395
00396
       /* Compute A^T B A... */
00397
       if (transpose == 1) {
00398
          /* Compute B^1/2 A... */
00399
00400
          for (i = 0; i < m; i++)
           for (j = 0; j < n; j++)
00401
00402
              gsl_matrix_set(aux, i, j,
00403
                            gsl_vector_get(b, i) * gsl_matrix_get(a, i, j));
00404
00405
          /* Compute A^T B A = (B^1/2 A)^T (B^1/2 A)...*/
00406
         gsl_blas_dgemm(CblasTrans, CblasNoTrans, 1.0, aux, aux, 0.0, c);
00407
00408
       /* Compute A B A^T... */
else if (transpose == 2) {
00409
00410
00411
00412
          /* Compute A B^1/2... */
00413
         for (i = 0; i < m; i++)
00414
           for (j = 0; j < n; j++)
00415
             gsl_matrix_set(aux, i, j,
00416
                             gsl_matrix_get(a, i, j) * gsl_vector_get(b, j));
00417
00418
          /* Compute A B A^T = (A B^1/2) (A B^1/2)^T... */
00419
         gsl_blas_dgemm(CblasNoTrans, CblasTrans, 1.0, aux, aux, 0.0, c);
00420
00421
00422
       /* Free... */
       gsl_matrix_free(aux);
00424 }
00425
00427
00428 void optimal estimation (
       ret_t * ret,
ctl_t * ctl,
00429
00430
00431
        obs_t * obs_meas,
00432
       obs_t * obs_i,
       atm_t * atm_apr,
00433
00434
       atm_t * atm_i) {
00435
00436
       static int ipa[N], iqa[N];
00437
00438
       gsl_matrix *a, *auxnm, *corr, *cov, *gain, *k_i, *s_a_inv;
00439
       gsl_vector *b, *dx, *dy, *sig_eps_inv, *sig_formod, *sig_noise,
00440
         *x_a, *x_i, *x_step, *y_aux, *y_i, *y_m;
00441
00442
       FILE *out;
00443
00444
       char filename[LEN];
00445
       double chisq, chisq_old, disq = 0, lmpar = 0.001;
00446
00447
00448
       int ig, ip, it = 0, it2, iw;
00449
00450
       size_t i, j, m, n;
00451
00452
00453
          Initialize...
00454
00455
00456
        /* Get sizes... */
00457
       m = obs2y(ctl, obs_meas, NULL, NULL, NULL);
       n = atm2x(ctl, atm\_apr, NULL, iqa, ipa);
00458
       if (m <= 0 || n <= 0)
00459
00460
         ERRMSG("Check problem definition!");
00461
00462
        /* Write info... */
       00463
00464
00465
               (int) m. (int) n.
               (double) (3 * m * n + 4 * n * n + 8 * m +
00466
                         8 * n) * sizeof(double) / 1024. / 1024.,
00467
               (double) (5 * sizeof(atm_t) + 3 * sizeof(obs_t) + 2 * N * sizeof(int)) / 1024. / 1024.);
00468
00469
00470
00471
       /* Allocate... */
00472
      a = gsl_matrix_alloc(n, n);
```

5.28 retrieval.c 271

```
cov = gsl_matrix_alloc(n, n);
00474
        k_i = gsl_matrix_alloc(m, n);
00475
        s_a_inv = gsl_matrix_alloc(n, n);
00476
00477
        b = gsl_vector_alloc(n);
       dx = gsl_vector_alloc(n);
dy = gsl_vector_alloc(m);
00478
00480
        sig_eps_inv = gsl_vector_alloc(m);
00481
        sig_formod = gsl_vector_alloc(m);
00482
        sig_noise = gsl_vector_alloc(m);
00483
        x_a = gsl_vector_alloc(n);
        x_i = gsl_vector_alloc(n);
00484
00485
        x step = gsl vector alloc(n);
        y_aux = gsl_vector_alloc(m);
00486
00487
        y_i = gsl_vector_alloc(m);
00488
       y_m = gsl_vector_alloc(m);
00489
00490
        /* Set initial state... */
00491
        copy_atm(ctl, atm_i, atm_apr, 0);
        copy_obs(ctl, obs_i, obs_meas, 0);
00492
00493
        formod(ctl, atm_i, obs_i);
00494
00495
        /* Set state vectors and observation vectors... */
00496
       atm2x(ctl, atm_apr, x_a, NULL, NULL);
00497
        atm2x(ctl, atm_i, x_i, NULL, NULL);
        obs2y(ctl, obs_meas, y_m, NULL, NULL);
00498
00499
        obs2y(ctl, obs_i, y_i, NULL, NULL);
00500
00501
       /* Set inverse a priori covariance S_a^-1... */
       00502
00503
00504
00505
        matrix_invert(s_a_inv);
00506
        /* Get measurement errors... */
00507
00508
        set_cov_meas(ret, ctl, obs_meas, sig_noise, sig_formod, sig_eps_inv);
00509
00510
        /* Create cost function file... */
00511
        sprintf(filename, "%s/costs.tab", ret->dir);
00512
          (!(out = fopen(filename, "w")))
00513
         ERRMSG("Cannot create cost function file!");
00514
00515
       /* Write header... */
00516
       fprintf(out,
                "# $1 = iteration number n"
00517
00518
                "# $2 = normalized cost function n"
00519
                "# $3 = number of measurements \n"
                "# $4 = number of state vector elements \n\n");
00520
00521
       /* Determine dx = x_i - x_a and dy = y - F(x_i) ... */
00522
00523
       gsl_vector_memcpy(dx, x_i);
00524
        gsl_vector_sub(dx, x_a);
00525
        gsl_vector_memcpy(dy, y_m);
00526
       gsl_vector_sub(dy, y_i);
00527
00528
        /* Compute cost function... */
       chisq = cost_function(dx, dy, s_a_inv, sig_eps_inv);
00530
       /* Write info... */
printf("it= %d / chi^2/m= %g\n", it, chisq);
00531
00532
00533
00534
        /* Write to cost function file... */
00535
       fprintf(out, "%d %g %d %d\n", it, chisq, (int) m, (int) n);
00536
00537
        /* Compute initial kernel... */
00538
       kernel(ctl, atm_i, obs_i, k_i);
00539
00540
00541
          Levenberg-Marguardt minimization...
00543
        /* Outer loop... */
00544
00545
        for (it = 1; it <= ret->conv_itmax; it++) {
00546
00547
          /* Store current cost function value... */
00548
         chisq_old = chisq;
00549
00550
          /\star Compute kernel matrix K_i... \star/
00551
         if (it > 1 && it % ret->kernel_recomp == 0)
           kernel(ctl, atm_i, obs_i, k_i);
00552
00553
          /* Compute K_i^T * S_eps^{-1} * K_i ... */
00555
          if (it == 1 || it % ret->kernel_recomp == 0)
00556
            matrix_product(k_i, sig_eps_inv, 1, cov);
00557
          /* Determine b = K_i^T * S_eps^{-1} * dy - S_a^{-1} * dx ... */
00558
00559
          for (i = 0; i < m; i++)
```

```
gsl_vector_set(y_aux, i, gsl_vector_get(dy, i)
00561
                              * gsl_pow_2(gsl_vector_get(sig_eps_inv, i)));
          gsl_blas_dgemv(CblasTrans, 1.0, k_i, y_aux, 0.0, b);
00562
00563
          gsl\_blas\_dgemv(CblasNoTrans, -1.0, s\_a\_inv, dx, 1.0, b);
00564
00565
           /* Inner loop... */
          for (it2 = 0; it2 < 20; it2++) {
00566
00567
00568
             /* Compute A = (1 + lmpar) * S_a^{-1} + K_i^T * S_eps^{-1} * K_i ... */
            gsl_matrix_memcpy(a, s_a_inv);
gsl_matrix_scale(a, 1 + lmpar);
00569
00570
00571
            gsl_matrix_add(a, cov);
00572
00573
             /* Solve A * x_step = b by means of Cholesky decomposition... */
00574
             gsl_linalg_cholesky_decomp(a);
00575
            gsl_linalg_cholesky_solve(a, b, x_step);
00576
00577
             /* Update atmospheric state... */
            gsl_vector_add(x_i, x_step);
00579
             copy_atm(ctl, atm_i, atm_apr, 0);
00580
             copy_obs(ctl, obs_i, obs_meas, 0);
00581
            x2atm(ctl, x_i, atm_i);
00582
             /* Check atmospheric state... */
00583
00584
             for (ip = 0; ip < atm_i->np; ip++) {
              atm_i \rightarrow p[ip] = GSL_MIN(GSL_MAX(atm_i \rightarrow p[ip], 5e-7), 5e4);
00585
00586
               atm_i \rightarrow t[ip] = GSL_MIN(GSL_MAX(atm_i \rightarrow t[ip], 100), 400);
00587
               for (ig = 0; ig < ctl->ng; ig++)
00588
                atm_i -> q[ig][ip] = GSL_MIN(GSL_MAX(atm_i -> q[ig][ip], 0), 1);
               for (iw = 0; iw < ctl->nw; iw++)
00589
00590
                atm_i \rightarrow k[iw][ip] = GSL_MAX(atm_i \rightarrow k[iw][ip], 0);
00591
00592
00593
             /\star Forward calculation... \star/
00594
             formod(ctl, atm_i, obs_i);
00595
            obs2y(ctl, obs_i, y_i, NULL, NULL);
00596
             /* Determine dx = x_i - x_a and dy = y - F(x_i) ... */
00598
            gsl_vector_memcpy(dx, x_i);
00599
             gsl_vector_sub(dx, x_a);
00600
             gsl_vector_memcpy(dy, y_m);
00601
             gsl_vector_sub(dy, y_i);
00602
00603
             /* Compute cost function... */
            chisq = cost_function(dx, dy, s_a_inv, sig_eps_inv);
00604
00605
00606
             /* Modify Levenberg-Marquardt parameter... */
00607
            if (chisq > chisq_old) {
              lmpar *= 10;
00608
00609
               qsl_vector_sub(x_i, x_step);
00610
            } else {
00611
               lmpar /= 10;
00612
              break;
00613
            }
00614
00615
00616
           /* Write info... */
00617
          printf("it= %d / chi^2/m= %g\n", it, chisq);
00618
          /* Write to cost function file... */ fprintf(out, "%d %g %d %d\n", it, chisq, (int) m, (int) n);
00619
00620
00621
00622
           /* Get normalized step size in state space... */
00623
          gsl_blas_ddot(x_step, b, &disq);
00624
          disq /= (double) n;
00625
00626
          /* Convergence test... */
if ((it == 1 || it % ret->kernel_recomp == 0) && disq < ret->
00627
      conv_dmin)
00628
            break;
00629
00630
00631
        /* Close cost function file... */
00632
        fclose(out);
00633
00634
        /* Store results... */
        00635
00636
00637
00638
00639
00640
00641
           Analysis of retrieval results...
00642
00643
00644
        /\star Check if error analysis is requested... \star/
00645
        if (ret->err ana) {
```

5.28 retrieval.c 273

```
/* Allocate... */
00647
00648
                            auxnm = gsl_matrix_alloc(n, m);
00649
                            corr = gsl_matrix_alloc(n, n);
00650
                            gain = gsl_matrix_alloc(n, m);
00651
00652
                            /* Compute inverse retrieval covariance...
00653
                                   cov^{-1} = S_a^{-1} + K_i^T * S_eps^{-1} * K_i */
00654
                            matrix_product(k_i, sig_eps_inv, 1, cov);
00655
                            gsl_matrix_add(cov, s_a_inv);
00656
00657
                            /* Compute retrieval covariance... */
00658
                           matrix_invert(cov);
00659
                            write_matrix(ret->dir, "matrix_cov_ret.tab", ctl, cov,
                           atm_i, obs_i, "x", "x", "r");
write_stddev("total", ret, ctl, atm_i, cov);
00660
00661
00662
00663
                             /* Compute correlation matrix... */
00664
                            for (i = 0; i < n; i++)</pre>
                               for (j = 0; j < n; j++)
00666
                                      gsl_matrix_set(corr, i, j, gsl_matrix_get(cov, i, j)
                           00667
00668
00669
00670
00671
00672
                           /* Compute gain matrix...
                            G = cov * K^T * S_eps^{-1} */
for (i = 0; i < n; i++)
00673
00674
                                 for (j = 0; j < m; j++)
00675
                                      gsl_matrix_set(auxnm, i, j, gsl_matrix_get(k_i, j, i)
    * gsl_pow_2(gsl_vector_get(sig_eps_inv, j)));
00676
00677
00678
                            gsl_blas_dgemm(CblasNoTrans, CblasNoTrans, 1.0, cov, auxnm, 0.0, gain);
00679
                            write_matrix(ret->dir, "matrix_gain.tab", ctl, gain,
                                                               atm_i, obs_i, "x", "y", "c");
00680
00681
                            /* Compute retrieval error due to noise... */
00682
                           matrix_product(gain, sig_noise, 2, a);
write_stddev("noise", ret, ctl, atm_i, a);
00683
00684
00685
00686
                            /\star Compute retrieval error % \left( 1\right) =\left( 1\right) +\left( 1\right
                           matrix_product(gain, sig_formod, 2, a);
write_stddev("formod", ret, ctl, atm_i, a);
00687
00688
00689
00690
                            /* Compute averaging kernel matrix
00691
00692
                            gsl_blas_dgemm(CblasNoTrans, CblasNoTrans, 1.0, gain, k_i, 0.0, a);
                            00693
00694
00695
00696
                            /* Analyze averaging kernel matrix... */
00697
                           analyze_avk(ret, ctl, atm_i, iqa, ipa, a);
00698
00699
                            /* Free... */
00700
                           gsl_matrix_free(auxnm);
00701
                            gsl_matrix_free(corr);
00702
                            gsl_matrix_free(gain);
00703
00704
00705
00706
                             Finalize...
00707
00708
00709
                      gsl_matrix_free(a);
00710
                      gsl_matrix_free(cov);
00711
                      gsl_matrix_free(k_i);
00712
                      gsl_matrix_free(s_a_inv);
00713
00714
                      gsl vector free(b):
                      gsl_vector_free(dx);
00716
                      gsl_vector_free(dy);
00717
                      gsl_vector_free(sig_eps_inv);
00718
                      gsl_vector_free(sig_formod);
00719
                      gsl_vector_free(sig_noise);
00720
                      gsl vector free(x a);
00721
                      gsl_vector_free(x_i);
00722
                      gsl_vector_free(x_step);
00723
                      gsl_vector_free(y_aux);
00724
                      gsl_vector_free(y_i);
00725
                      gsl_vector_free(y_m);
00726 }
00727
00729
00730 void read_ret(
00731 int argc,
00732
                   char *argv[].
```

```
00733
         ctl_t * ctl,
00734
         ret_t * ret) {
00735
00736
         int id, ig, iw;
00737
00738
          /* Iteration control... */
00739
         ret->kernel_recomp
         (int) scan_ctl(argc, argv, "KERNEL_RECOMP", -1, "3", NULL);
ret->conv_itmax = (int) scan_ctl(argc, argv, "CONV_ITMAX", -1, "30", NULL);
ret->conv_dmin = scan_ctl(argc, argv, "CONV_DMIN", -1, "0.1", NULL);
00740
00741
00742
00743
00744
         /* Error analysis... */
         ret->err_ana = (int) scan_ctl(argc, argv, "ERR_ANA", -1, "1", NULL);
00745
00746
00747
          for (id = 0; id < ctl->nd; id++)
00748
           ret->err_formod[id] = scan_ctl(argc, argv, "ERR_FORMOD", id, "0", NULL);
00749
00750
         for (id = 0; id < ctl->nd; id++)
           ret->err_noise[id] = scan_ctl(argc, argv, "ERR_NOISE", id, "0", NULL);
00751
00752
         ret->err_press = scan_ctl(argc, argv, "ERR_PRESS", -1, "0", NULL);
ret->err_press_cz = scan_ctl(argc, argv, "ERR_PRESS_CZ", -1, "-999", NULL);
ret->err_press_ch = scan_ctl(argc, argv, "ERR_PRESS_CH", -1, "-999", NULL);
00753
00754
00755
00756
         ret->err_temp = scan_ctl(argc, argv, "ERR_TEMP", -1, "0", NULL);
ret->err_temp_cz = scan_ctl(argc, argv, "ERR_TEMP_CZ", -1, "-999", NULL);
ret->err_temp_ch = scan_ctl(argc, argv, "ERR_TEMP_CH", -1, "-999", NULL);
00757
00758
00759
00760
00761
          for (ig = 0; ig < ctl->ng; ig++) {
          ret->err_q[ig] = scan_ctl(argc, argv, "ERR_Q", ig, "0", NULL);
ret->err_q_cz[ig] = scan_ctl(argc, argv, "ERR_Q_CZ", ig, "-999", NULL);
ret->err_q_ch[ig] = scan_ctl(argc, argv, "ERR_Q_CH", ig, "-999", NULL);
00762
00763
00764
00765
00766
00767
         for (iw = 0; iw < ctl->nw; iw++) {
           ret->err_k[iw] = scan_ctl(argc, argv, "ERR_K", iw, "0", NULL);
ret->err_k_cz[iw] = scan_ctl(argc, argv, "ERR_K_CZ", iw, "-999", NULL);
ret->err_k_ch[iw] = scan_ctl(argc, argv, "ERR_K_CH", iw, "-999", NULL);
00768
00769
00770
00771
00772 }
00773
00775
00776 void set_cov_apr(
00777
         ret_t * ret,
00778
          ctl_t * ctl,
00779
         atm_t * atm,
00780
         int *iqa,
00781
         int *ipa,
00782
         gsl matrix * s a) {
00783
00784
         gsl_vector *x_a;
00785
00786
         double ch, cz, rho, x0[3], x1[3];
00787
00788
         int iq, iw;
00789
00790
         size_t i, j, n;
00791
         /* Get sizes... */
00792
00793
         n = s_a->size1;
00794
00795
          /* Allocate... */
00796
         x_a = gsl_vector_alloc(n);
00797
00798
          /* Get sigma vector... */
00799
          atm2x(ctl, atm, x_a, NULL, NULL);
for (i = 0; i < n; i++) {</pre>
00800
           if (iga[i] == IDXP)
00801
00802
              gsl_vector_set(x_a, i, ret->err_press / 100 * gsl_vector_get(x_a, i));
00803
                (iqa[i] == IDXT)
00804
              gsl_vector_set(x_a, i, ret->err_temp);
            for (ig = 0; ig < ctl->ng; ig++)
  if (iqa[i] == IDXQ(ig))
00805
00806
            00807
00808
00809
               if (iqa[i] == IDXK(iw))
00810
                 gsl_vector_set(x_a, i, ret->err_k[iw]);
00811
00812
00813
          /* Check standard deviations... */
00814
          for (i = 0; i < n; i++)</pre>
00815
           if (gsl_pow_2(gsl_vector_get(x_a, i)) <= 0)</pre>
00816
               ERRMSG("Check a priori data (zero standard deviation)!");
00817
00818
         /* Initialize diagonal covariance... */
00819
         gsl matrix set zero(s a);
```

5.28 retrieval.c 275

```
for (i = 0; i < n; i++)</pre>
00821
          gsl_matrix_set(s_a, i, i, gsl_pow_2(gsl_vector_get(x_a, i)));
00822
00823
        /\star Loop over matrix elements... \star/
00824
        for (i = 0; i < n; i++)
  for (j = 0; j < n; j++)</pre>
00825
            if (i != j && iqa[i] == iqa[j]) {
00827
00828
               /* Initialize... */
00829
               cz = ch = 0;
00830
00831
               /\star Set correlation lengths for pressure... \star/
00832
               if (iqa[i] == IDXP) {
00833
                cz = ret->err_press_cz;
00834
                 ch = ret->err_press_ch;
00835
00836
00837
               /\star Set correlation lengths for temperature... \star/
               if (iqa[i] == IDXT) {
00838
00839
                cz = ret->err_temp_cz;
00840
                 ch = ret->err_temp_ch;
00841
00842
00843
               /\!\star Set correlation lengths for volume mixing ratios... \!\star/\!
               for (ig = 0; ig < ctl->ng; ig++)
  if (iqa[i] == IDXQ(ig)) {
00844
00846
                   cz = ret->err_q_cz[ig];
                   ch = ret->err_q_ch[ig];
00847
00848
                 }
00849
00850
               /* Set correlation lengths for extinction... */
               for (iw = 0; iw < ctl->nw; iw++)
  if (iqa[i] == IDXK(iw)) {
00851
00852
00853
                   cz = ret->err_k_cz[iw];
                   ch = ret->err_k_ch[iw];
00854
00855
00856
               /* Compute correlations... */
00858
               if (cz > 0 && ch > 0) {
00859
00860
                 /* Get Cartesian coordinates... */
                 geo2cart(0, atm->lon[ipa[i]], atm->lat[ipa[i]], x0);
geo2cart(0, atm->lon[ipa[j]], atm->lat[ipa[j]], x1);
00861
00862
00863
                  /* Compute correlations... */
00865
                 rho =
00866
                   exp(-DIST(x0, x1) / ch -
                        fabs(atm->z[ipa[i]] - atm->z[ipa[j]]) / cz);
00867
00868
                 /* Set covariance... */
gsl_matrix_set(s_a, i, j, gsl_vector_get(x_a, i)
00869
00870
00871
                                  * gsl_vector_get(x_a, j) * rho);
00872
00873
            }
00874
00875
        /* Free... */
        gsl_vector_free(x_a);
00877 }
00878
00880
00881 void set cov meas(
00882
        ret_t * ret,
        ctl_t * ctl,
00883
00884
        obs_t * obs,
        gsl_vector * sig_noise,
gsl_vector * sig_formod,
00885
00886
        gsl_vector * sig_eps_inv) {
00887
00888
        static obs_t obs_err;
00890
00891
        int id, ir;
00892
00893
        size t i, m;
00894
00895
        /* Get size... */
00896
        m = sig_eps_inv->size;
00897
00898
        /\star Noise error (always considered in retrieval fit)... \star/
        copy_obs(ctl, &obs_err, obs, 1);
for (ir = 0; ir < obs_err.nr; ir++)
00899
00900
          for (id = 0; id < ctl->nd; id++)
00901
00902
             obs_err.rad[id][ir]
00903
                 (gsl_finite(obs->rad[id][ir]) ? ret->err_noise[id] : GSL_NAN);
00904
        obs2y(ctl, &obs_err, sig_noise, NULL, NULL);
00905
00906
        /* Forward model error (always considered in retrieval fit)... */
```

```
copy_obs(ctl, &obs_err, obs, 1);
00908
        for (ir = 0; ir < obs_err.nr; ir++)</pre>
        for (id = 0; id < ctl->nd; id++)
00909
           obs_err.rad[id][ir]
00910
              = fabs(ret->err_formod[id] / 100 * obs->rad[id][ir]);
00911
        obs2y(ctl, &obs_err, sig_formod, NULL, NULL);
00912
00913
00914
        /* Total error... */
00915
        for (i = 0; i < m; i++)</pre>
00916
         gsl_vector_set(sig_eps_inv, i,
                          1 / sqrt(gsl_pow_2(gsl_vector_get(sig_noise, i))
00917
00918
                                   + gsl_pow_2(gsl_vector_get(sig_formod, i))));
00919
00920
        /* Check standard deviations... */
00921
        for (i = 0; i < m; i++)</pre>
00922
         if (gsl_vector_get(sig_eps_inv, i) <= 0)</pre>
00923
            ERRMSG("Check measurement errors (zero standard deviation)!");
00924 }
00925
00927
00928 void write_stddev(
00929
        const char *quantity,
       ret_t * ret,
ctl_t * ctl,
atm_t * atm,
00930
00931
00932
00933
        gsl_matrix * s) {
00934
00935
        static atm_t atm_aux;
00936
00937
        gsl vector *x aux;
00938
00939
        char filename[LEN];
00940
00941
        size_t i, n;
00942
00943
        /* Get sizes... */
00944
        n = s \rightarrow size1;
00945
00946
        /* Allocate... */
00947
        x_aux = gsl_vector_alloc(n);
00948
00949
        /* Compute standard deviation... */
00950
        for (i = 0; i < n; i++)
00951
         gsl_vector_set(x_aux, i, sqrt(gsl_matrix_get(s, i, i)));
00952
00953
        /* Write to disk... */
00954
        copy_atm(ctl, &atm_aux, atm, 1);
        x2atm(ctl, x_aux, &atm_aux);
sprintf(filename, "atm_err_%s.tab", quantity);
write_atm(ret->dir, filename, ctl, &atm_aux);
00955
00956
00957
00958
00959
        /* Free... */
00960
       gsl_vector_free(x_aux);
00961 }
```

# 5.29 time2jsec.c File Reference

Convert date to Julian seconds.

## **Functions**

• int main (int argc, char \*argv[])

# 5.29.1 Detailed Description

Convert date to Julian seconds.

Definition in file time2jsec.c.

5.30 time2jsec.c 277

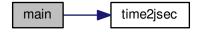
#### 5.29.2 Function Documentation

## 5.29.2.1 int main ( int argc, char \* argv[])

Definition at line 27 of file time2jsec.c.

```
00029
00030
00031
        double jsec, remain;
00032
00033
        int day, hour, min, mon, sec, year;
00034
00035
         /* Check arguments... */
00036
          ERRMSG("Give parameters: <year> <mon> <day> <hour> <min> <sec> <remain>");
00037
00038
00039
        /* Read arguments... */
00040
        year = atoi(argv[1]);
00041
        mon = atoi(argv[2]);
00042
        day = atoi(argv[3]);
00043
        hour = atoi(argv[4]);
00044
        min = atoi(argv[5]);
sec = atoi(argv[6]);
00045
00046
        remain = atof(argv[7]);
00047
00048
        time2jsec(year, mon, day, hour, min, sec, remain, &jsec);
printf("%.2f\n", jsec);
00049
00050
00051
00052
        return EXIT_SUCCESS;
00053 }
```

Here is the call graph for this function:



# 5.30 time2jsec.c

```
00001 /*
00002
        This file is part of JURASSIC.
00003
00004
         JURASSIC is free software: you can redistribute it and/or modify
00005
         it under the terms of the GNU General Public License as published by
00006
         the Free Software Foundation, either version 3 of the License, or
00007
         (at your option) any later version.
80000
00009
         {\tt JURASSIC} is distributed in the hope that it will be useful,
         but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00010
00011
00012
         GNU General Public License for more details.
00013
00014
         You should have received a copy of the GNU General Public License
00015
        along with JURASSIC. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.
00016
00017
         Copright (C) 2003-2015 Forschungszentrum Juelich GmbH
00018 */
00019
00025 #include "jurassic.h"
00026
00027 int main(
00028
      int argc,
00029
        char *argv[]) {
```

```
00030
00031
         double jsec, remain;
00032
00033
00034
         int day, hour, min, mon, sec, year;
00035
          /* Check arguments... */
00036
          if (argc < 8)
00037
            ERRMSG("Give parameters: <year> <mon> <day> <hour> <min> <sec> <remain>");
00038
         /* Read arguments... */
year = atoi(argv[1]);
mon = atoi(argv[2]);
day = atoi(argv[3]);
00039
00041
00042
         hour = atoi(argv[4]);
min = atoi(argv[5]);
sec = atoi(argv[6]);
00043
00044
00045
00046
         remain = atof(argv[7]);
00047
         /* Convert... */
00048
         time2jsec(year, mon, day, hour, min, sec, remain, &jsec);
printf("%.2f\n", jsec);
00049
00050
00051
00052 return EXIT_SUCCESS;
00053 }
```

# Index

analyze_avk	hydz, 7
retrieval.c, 253	nd, <b>7</b>
analyze_avk_quantity	ng, <mark>6</mark>
retrieval.c, 254	nu, 7
atm2x	nw, <b>7</b>
jurassic.c, 37	rayds, 8
jurassic.h, 166	raydz, 8
atm2x_help	refrac, 8
jurassic.c, 38	retk_zmax, 9
jurassic.h, 166	retk_zmin, 9
atm_t, 3	retp zmax, 8
k, 5	retp_zmin, 8
	• —
lat, 4	retq_zmax, 9
lon, 4	retq_zmin, 9
np, 4	rett_zmax, 8
p, 4	rett_zmin, 8
q, 5	tblbase, 7
t, 4	window, 7
time, 4	write_bbt, 9
z, 4	write_matrix, 9
	ctm_co2
brightness	ctl_t, <b>7</b>
jurassic.c, 38	ctm_h2o
jurassic.h, 166	ctl_t, 7
brightness.c, 19	ctm_n2
main, 20	ctl_t, 7
	ctm o2
call_formod	ctl_t, 8
formod.c, 23	ctmco2
cart2geo	jurassic.c, 48
jurassic.c, 38	jurassic.h, 176
jurassic.h, 167	ctmh2o
climatology	jurassic.c, 58
jurassic.c, 39	jurassic.h, 186
jurassic.h, 167	ctmn2
climatology.c, 21	
main, 21	jurassic.c, 70
conv dmin	jurassic.h, 198
ret_t, 15	ctmo2
conv itmax	jurassic.c, 71
ret_t, 15	jurassic.h, 199
copy atm	مانہ
jurassic.c, 72	dir
jurassic.h, 200	ret_t, 15
copy_obs	ds
• • —	los_t, 11
jurassic.c, 73	a mailth a m
jurassic.h, 201	emitter
cost_function	ctl_t, 6
retrieval.c, 254	eps
ctl_t, 5	tbl_t, 19
ctm_co2, 7	err_ana
ctm_h2o, 7	ret_t, 15
ctm_n2, 7	err_formod
ctm_o2, 8	ret_t, 16
emitter, 6	err_k
fov, 8	ret_t, 17

err_k_ch	hydrostatic.c, 30
ret_t, 17	main, 30
err_k_cz	hydz
ret_t, 17	ctl_t, 7
err_noise	
ret_t, 16	idx2name
err_press	jurassic.c, <mark>82</mark>
ret_t, 16	jurassic.h, 210
err_press_ch	init_tbl
ret_t, 16	jurassic.c, 82
err press cz	jurassic.h, 210
ret_t, 16	interpolate.c, 32
err_q	main, 32
ret_t, 16	intpol_atm
err q ch	jurassic.c, 84
— <del>—</del>	jurassic.h, 212
ret_t, 17	intpol tbl
err_q_cz	jurassic.c, 84
ret_t, 17	jurassic.h, 212
err_temp	intpol_tbl_eps
ret_t, 16	
err_temp_ch	jurassic.c, 86
ret_t, 16	jurassic.h, 214
err_temp_cz	intpol_tbl_u
ret_t, 16	jurassic.c, 87
	jurassic.h, 215
find_emitter	. 01,
jurassic.c, 73	jsec2time
jurassic.h, 201	jurassic.c, 87
formod	jurassic.h, 215
jurassic.c, 73	jsec2time.c, 34
jurassic.h, 202	main, 34
formod.c, 23	jurassic.c, 35
call_formod, 23	atm2x, 37
main, 25	atm2x_help, 38
formod_continua	brightness, 38
jurassic.c, 75	cart2geo, <mark>38</mark>
jurassic.h, 203	climatology, 39
formod_fov	copy_atm, 72
jurassic.c, 76	copy_obs, 73
jurassic.b, 70 jurassic.h, 204	ctmco2, 48
-	ctmh2o, 58
formod_pencil	ctmn2, 70
jurassic.c, 78	ctmo2, 71
jurassic.h, 206	find_emitter, 73
formod_srcfunc	formod, 73
jurassic.c, 80	formod continua, 75
jurassic.h, 208	<del>-</del>
fov	formod_fov, 76
ctl_t, 8	formod_pencil, 78
	formod_srcfunc, 80
geo2cart	geo2cart, 80
jurassic.c, 80	gravity, 80
jurassic.h, 208	hydrostatic, 80
gravity	idx2name, 82
jurassic.c, 80	init_tbl, 82
jurassic.h, 208	intpol_atm, 84
	intpol_tbl, 84
hydrostatic	intpol_tbl_eps, 86
jurassic.c, 80	intpol_tbl_u, 87
jurassic.h, 208	jsec2time, 87
-	•

	kernel, 88		read_matrix, 224
	locate, 90		read_obs, 224
	locate_tbl, 91		read_shape, 225
	obs2y, 91		refractivity, 226
	planck, 91		scan_ctl, 226
	raytrace, 91		tangent_point, 227
	read_atm, 94		time2jsec, 228
	read ctl, 95		timer, 228
	read_matrix, 96		write_atm, 229
	read_obs, 96		write matrix, 229
	read_shape, 97		write obs, 232
	refractivity, 98		x2atm, 233
	scan_ctl, 98		x2atm_help, 233
	tangent point, 99		y2obs, 233
	time2jsec, 100		)=000, <b>=</b> 00
	-	k	
	timer, 100		atm t, 5
	write_atm, 101		los t, 11
	write_matrix, 101	kern	— <i>'</i>
	write_obs, 104		jurassic.c, 88
	x2atm, 105		jurassic.h, 216
	x2atm_help, 105	kern	el.c, 241
	y2obs, 105	KCIII	main, 241
juras	sic.h, 163	kern	el_recomp
	atm2x, 166	KCIII	ret_t, 15
	atm2x_help, 166		101_1, 10
	brightness, 166	lat	
	cart2geo, 167		atm t, 4
	climatology, 167		los_t, 10
	copy_atm, 200	limb	.c, 244
	copy_obs, 201		main, 245
	ctmco2, 176	loca	,
	ctmh2o, 186	ioca	jurassic.c, 90
	ctmn2, 198		jurassic.h, 218
	ctmo2, 199	locat	te tbl
	find_emitter, 201	ioca	jurassic.c, 91
	formod, 202		jurassic.h, 219
	formod_continua, 203	lon	jurassic.ri, 215
	formod_fov, 204	1011	atm t, 4
	formod_pencil, 206		los t, 10
	formod_srcfunc, 208	los	<del>-</del> :
	geo2cart, 208	103_	เ, 9 ds, 11
	gravity, 208		k, 11
	hydrostatic, 208		lat, 10
	idx2name, 210		lon, 10
	init tbl, 210		np, 10
	intpol atm, 212		p, 11
	intpol tbl, 212		q, 11
	intpol_tbl_eps, 214		•
	intpol_tbl_u, 215		t, 11 tsurf, 11
	jsec2time, 215		u, 11
	kernel, 216		
	locate, 218		z, 10
	locate_tbl, 219	mair	1
	obs2y, 219	mail	brightness.c, 20
	planck, 219		climatology.c, 21
	raytrace, 219		formod.c, 25
	read atm, 222		
	<del>-</del> · ·		hydrostatic.c, 30
	read_ctl, 223		interpolate.c, 32

jsec2time.c, 34	p
kernel.c, 241	atm_t, 4
limb.c, 245	los_t, 11
nadir.c, 247	tbl_t, 18
planck.c, 248	planck
raytrace.c, 250	jurassic.c, 91
retrieval.c, 264	jurassic.h, 219
time2jsec.c, 277	planck.c, 248
matrix_invert	main, 248
retrieval.c, 255	
matrix_product	q
retrieval.c, 255	atm_t, 5
,	los_t, 11
nadir.c, 246	
main, 247	rad
nd	obs_t, 14
ctl_t, 7	rayds
ng	ctl_t, 8
ctl t, 6	raydz
np	ctl_t, 8
atm_t, 4	raytrace
los_t, 10	jurassic.c, 91
tbl_t, 18	jurassic.h, 219
nr	raytrace.c, 250
obs_t, 12	main, 250
nt	read_atm
tbl_t, 18	jurassic.c, 94
nu	jurassic.h, 222
ctl_t, 7	read_ctl jurassic.c, 95
tbl_t, 18	jurassic.h, 223
nw	read matrix
ctl_t, 7	jurassic.c, 96
	jurassic.h, 224
obs2y	read_obs
jurassic.c, 91	jurassic.c, 96
jurassic.h, 219	jurassic.t, 30
obs_t, 12	read_ret
nr, 12	retrieval.c, 260
obslat, 13	read_shape
obslon, 13	jurassic.c, 97
obsz, 13	jurassic.h, 225
rad, 14	refrac
tau, 14	ctl_t, 8
time, 12	refractivity
tplat, 13	jurassic.c, 98
tplon, 13	jurassic.h, 226
tpz, 13	ret_t, 14
vplat, 13	conv_dmin, 15
vplon, 13	conv_itmax, 15
vpz, 13	dir, 15
obslat	err_ana, 15
obs_t, 13	err_formod, 16
obslon	err_k, 17
obs_t, 13	err_k_ch, 17
obsz	err_k_cz, 17
obs_t, 13	err_noise, 16
optimal_estimation	err_press, 16
retrieval.c, 256	err_press_ch, 16

err_press_cz, 16	eps, 19
err_q, 16	np, 18
err_q_ch, 17	nt, 18
err_q_cz, 17	nu, 18
err_temp, 16	p, 18
err_temp_ch, 16	sr, 19
err_temp_cz, 16	st, 19
kernel_recomp, 15	t, 19
retk zmax	u, 19
 ctl_t, 9	tblbase
retk zmin	ctl_t, 7
ctl_t, 9	time
retp zmax	atm t, 4
ctl_t, 8	obs_t, 12
retp zmin	time2jsec
ctl_t, 8	
retq zmax	jurassic.c, 100 jurassic.h, 228
ctl_t, 9	•
retq_zmin	time2jsec.c, 276
ctl_t, 9	main, 277
retrieval.c, 252	timer
	jurassic.c, 100
analyze_avk, 253	jurassic.h, 228
analyze_avk_quantity, 254	tplat
cost_function, 254	obs_t, 13
main, 264	tplon
matrix_invert, 255	obs_t, 13
matrix_product, 255	tpz
optimal_estimation, 256	obs_t, 13
read_ret, 260	tsurf
set_cov_apr, 261	los_t, 11
set_cov_meas, 262	
write_stddev, 263	u
rett_zmax	los_t, 11
ctl_t, 8	tbl_t, 19
rett_zmin	
ctl_t, 8	vplat
	obs_t, 13
scan_ctl	vplon
jurassic.c, 98	obs_t, 13
jurassic.h, 226	vpz
set_cov_apr	obs_t, 13
retrieval.c, 261	000 <u>_</u> i, 10
set_cov_meas	window
retrieval.c, 262	ctl_t, 7
sr	write atm
tbl_t, 19	jurassic.c, 101
st	jurassic.t, 101
tbl_t, 19	write_bbt
t	ctl_t, 9
atm_t, 4	write_matrix
los_t, 11	ctl_t, 9
tbl_t, 19	jurassic.c, 101
tangent_point	jurassic.h, 229
jurassic.c, 99	write_obs
jurassic.h, 227	jurassic.c, 104
tau	jurassic.h, 232
obs_t, 14	write_stddev
tbl_t, 17	retrieval.c, 263

```
x2atm

jurassic.c, 105

jurassic.h, 233

x2atm_help

jurassic.c, 105

jurassic.h, 233

y2obs

jurassic.c, 105

jurassic.h, 233

z

atm_t, 4

los_t, 10
```